ISSN 0542 9951

9
1995

NAJSTARSZE PISMO INFORMATYCZNE W POLSCE – UKAZUJE SIĘ OD 1965 ROKU

BIBLIOTEKA ŚLĄSKA
Biblioteka Główna
w Katowicach
P 1877/95

informatyka

miesięcznik profesjonalistów

CZAS NA ZMIANY

Może nie wszyscy sobie to dobrze uświadamiają, ale „Informatyka" ma już 30 lat i jest najstarszym polskim (i jednym z najstarszych w Europie) pismem komputerowym. Co zaszło w ciągu tego czasu w informatyce, jakich przemian byliśmy świadkami – nikomu nie trzeba tłumaczyć; no, może tym tylko z najmłodszych, dla których świat zaczął się przedwczoraj.

W tym kontekście „Informatyka" jako pismo okazuje się – paradoksalnie – bytem niesłychanie konserwatywnym. W ciągu trzydziestu lat jedna ledwie zmiana tytułu (kto nie pamięta – niegdyś nazywaliśmy się drętwo „Elektroniczne maszyny matematyczne"), gdy zaś chodzi o układ graficzny strony, to różnice są zaniedbywalne nawet dla najlepszego fachowca żurnalistyki. Wygląda na to, że do wystroju pisma przywiązywano wagę wręcz minimalną...

Nawiasem mówiąc, spostrzeżenie to dotyczy nie tylko „Informatyki", ale większości polskich pism naukowych i popularnonaukowych. Trzeba było – koniecznie! – do tego zamulonego nieco stawiku samozadowolenia wpuścić parę komercyjnych szczupaków, żeby zaczął się jakiś ruch.

Zatem – czas na kolejne zmiany. Czas, by „środek przekazu sam stał się po trosze przekazem", jak chcą klasycy wiedzy o mass mediach. Od tego numeru okładka pisma będzie wyglądała podobnie jak ta, którą mają Państwo w ręce. Od dziś także bardzo wyraźnie oddzielamy część publikacyjną pisma od informacyjnej, tę ostatnią nieco wzbogacając. W tej części informacyjnej znajdą Państwo bogatsze formy dziennikarskie – nie tylko suche notatki, ale wszystko to, co w piśmie być może, i co potrafimy w naszym nielicznym (na razie, mam nadzieję) zespole sprokurować.

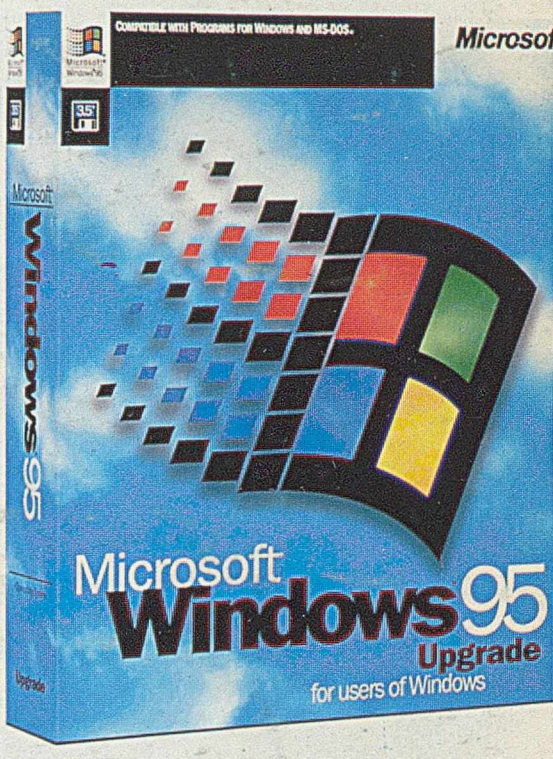
Jak Państwo zapewne bez trudu zauważą, troszeczkę zmieniamy także wnętrze pisma; będziemy starali się, by jego wystrój graficzny był przejrzystszy, bardziej konsekwentny i atrakcyjniejszy; na tyle naturalnie, na ile mamy swobody w istniejących warunkach poligraficznych.

Najważniejszą pozostaje jednak część publikacyjna. Chcemy utrzymać jej wysoki poziom i stricte naukowy charakter. Pragniemy, by opublikowanie artykułu na naszych łamach stało się prestiżowym elementem życia naukowego (zanim zacznie Autorom przynosić także takie profity materialne, na jakie zasługują...). Z tego powodu wprowadzamy zasadę imiennego rekomendowania tekstów przez samodzielnych pracowników nauki; taki rekomendowany artykuł nie będzie podlegał recenzowaniu wewnętrznemu, ewentualna adiustacja redakcyjna dotyczyć zaś będzie wyłącznie jego strony językowej. Samodzielni pracownicy nauki wszystkich dyscyplin są zatem uprawnieni do zalecenia redakcji druku tekstu, uważanego przez siebie za wartościowy (naturalnie, biorąc pod uwagę specyfikę „Informatyki"...); z drugiej strony, PT Autorzy mają prawo sami postarać się o czyjąś rekomendację i nadsyłać do redakcji artykuły, opatrzone stosownymi adnotacjami. Artykuły rekomendowane będą publikowane na czołowych miejscach pisma, w pierwszej kolejności.

Nie znaczy to, że zamykamy drogę na łamy „Informatyki" Autorom innych prac. Tyle, że prace nie rekomendowane będą podlegały recenzowaniu wewnętrznemu i opublikujemy je tylko wówczas, gdy uzyskają pozytywną ocenę przynajmniej jednego członka naszego kolegium. Zastrzegamy sobie – w wypadku tych prac – także prawo adiustacji merytorycznej; w uzgodnieniu z Autorem, ma się rozumieć.

Jest kwestią przypadkowego zbiegu okoliczności, że zmiany w „Informatyce" zbiegają się ze zmianami w... informatyce, tej ważniejszej, choć pisanej przez małe „i". Po długim wyczekiwaniu i licznych perypetiach wchodzi na światowy rynek system Microsoft Windows'95 i – przynajmniej dla niżej podpisanego – nie ulega wątpliwości, że od tego momentu świat komputerów osobistych (a pewno i nie tylko ten) będzie wyglądał nieco inaczej. Choć piszę te słowa na kilka tygodni przed zapowiedzianą premierą – jestem zupełnie pewien, że mam rację: tyle już opracowano dla nowego systemu aplikacji, tyle włożono weń pracy, pieniędzy i nadziei, że to się po prostu nie ma prawa Gatesowi nie udać. Zresztą, mój komputer już korzysta z kolejnej wersji beta tego systemu – i wszystko jest OK.

I dlatego właśnie obok tego tekstu mają Państwo w całej okazałości logo nowych „okienek". Chciałbym, by Państwo mocno skojarzyli zmiany w naszym piśmie z nową epoką w informatyce. Może to i nieskromne, ale życie uczy, że skromni sukcesy odnoszą nader rzadko. A ja chcę, by „Informatyka" odniosła u Państwa sukces: powagą, rzetelnością, szybkością informowania.



Liczę też na Państwa listy.

WYDAWNICTWO SIGMA NOT

informatyka

nr 9 1995
Wrzesień
Rok wydania XXXI

Miesięcznik
ISSN 0542-9951
INDEKS 36124

REDAGUJE ZESPÓŁ:
mgr Bogdan **MIŚ**
(redaktor naczelny)
mgr Krystyna
KARWICKA-RYCHLEWICZ
(zastępca redaktora naczelnego)
mgr Teresa **JABŁOŃSKA**
(sekretarz redakcji)
Alina **KLEPACZ**
(sekretariat)

KOLEGIUM REDAKCYJNE:
prof. dr hab. Leonard BOLC,
mgr inż. Piotr FUGLEWICZ,
prof. dr hab. Jan GOLIŃSKI,
dr inż. Zenon KULPA,
doc. dr inż. Jan MULAWKA,
prof. dr hab. Wojciech OLEJNICZAK,
mgr inż. Jan RYŻKO,
dr Witold STANISZKIS,
dr inż. Jacek STOCHLAK,
prof. dr hab. Maciej STOLARSKI,
dr hab. Zdzisław SZYJEWSKI,
prof. dr hab. inż.
Ryszard TADEUSIEWICZ,
prof. dr hab. Jan WĘGLARZ.

PRZEWODNICZĄCY
RADY PROGRAMOWEJ
prof. dr hab.
Juliusz Lech KULIKOWSKI

WYDAWCA:
Wydawnictwo Czasopism i Książek Technicznych SIGMA NOT Spółka z o.o.
ul. Ratuszowa 11
00-950 WARSZAWA
skrytka pocztowa 1004

Redakcja:
00-950 Warszawa,
ul. Ratuszowa 11, p. 628 i 644
skrytka pocztowa 1004
tel., fax: 619-11-61
tel.: 619-22-41 w. 159

Materiałów nie zamówionych redakcja nie zwraca

Autorzy artykułów proszeni są o przysyłanie tekstów na dyskietkach – w czystych kodach ASCII (Latin II, Mazovia) lub edytorach: Word 2, Word 6 – oraz odbitki na papierze.

W sprawach ogłoszeń prosimy zwracać się bezpośrednio do Redakcji lub Działu Reklamy i Marketingu
00-950 Warszawa
ul. Mazowiecka 12
telefon: 27-43-66
telefaks: 26-80-16

W numerze:

	Strona
Czas na zmiany – *Bogdan Miś*	I okł.
publikacje	
II Konferencja Systemy Czasu Rzeczywistego – *Zygmunt Mazur, Zbigniew Huzar*	1
Problemy jakości w projektowaniu oprogramowania systemów wbudowanych – *Krzysztof Sacha*	2
Obiektowe podejście do modelowania i analizy komputerowych systemów sterowania – *Janusz Górski, Bartosz Nowicki*	13
Więzy integralności a zarządzanie transakcjami w SQL-owych systemach baz danych czasu rzeczywistego – *Tadeusz Pankowski*	21
informacje	
Wiosenne Forum SRSO Łańsk '95. Świat systemów otwartych – *Krystyna Karwicka*	28
Inteligentne koncentratory MultiNet, czyli ta...ka ryba w Twojej sieci – *Przemysław Baszkiewicz*	29
ComNet Warsaw '95 – *Krystyna Karwicka*	30
APC dodaje gniazdo rozszerzeń do SmartUPS – *Przemysław Baszkiewicz*	30

W najbliższych numerach:

- Waldemar Wieczerzycki i Robert Wrembel charakteryzują perspektywy obiektowych baz danych.
- Halina Kwaśnicka omawia książkę J. Chromiec i E. Strzemiennej „Sztuczna inteligencja. Metody konstrukcji i analizy systemów eksperckich".
- Krzysztof Sacha charakteryzuje metody analizy i specyfikacji wymagań dla oprogramowania systemów czasu rzeczywistego.
- Zbigniew Fryźlewicz opisuje nowy mechanizm synchronizacji zadań wprowadzony w ramach modernizacji języka Ada.

Warunki prenumeraty na 1995 r.

Zamówienia na prenumeratę czasopism wydawanych przez Wydawnictwo SIGMA-NOT można składać w dowolnym terminie. Mogą one obejmować dowolny okres czasu, tzn. dotyczyć dowolnej liczby kolejnych zeszytów każdego czasopisma.

Zamawiający może otrzymywać zaprenumerowany przez siebie tytuł począwszy od następnego miesiąca po dokonaniu wpłaty. Zamówienia na zeszyty sprzed daty otrzymania wpłaty będą realizowane w miarę możliwości – z posiadanych zapasów magazynowych.

Warunkiem przyjęcia i realizacji zamówienia jest otrzymanie z banku potwierdzenia dokonania wpłaty przez prenumeratora. Dokument wpłaty jest równoznaczny ze złożeniem zamówienia.

Wpłat na prenumeratę można dokonywać na ogólnie dostępnych blankietach w Urzędach Pocztowych (przekazy pieniężne) lub Bankach (polecenie przelewu), przekazując środki pod adres:
Wydawnictwo SIGMA-NOT Spółka z o.o.
Zakład Kolportażu
00-716 Warszawa, skr. poczt. 1004
konto:
PBK S.A. III 0/Warszawa nr 370015-1573-139-11

Wpłaty na prenumeratę przyjmują także wszystkie urzędy pocztowe nadawczo-odbiorcze oraz doręczyciele na terenie całego kraju

Na blankiecie wpłaty należy czytelnie podać nazwę zamawianego czasopisma, liczbę zamawianych egzemplarzy, okres prenumeraty oraz własny adres.
Na życzenie prenumeratora, zgłoszone np. telefonicznie, Zakład Kolportażu ul. Bartycka 20, 00-950 Warszawa, (telefony: 40-30-86, 40-35-89 oraz 40-00-21 wew. 249, 293, 299) wysyła specjalne blankiety zamówień wraz z aktualną listą tytułów i cennikiem czasopism.

Odbiorcy zagraniczni mogą otrzymywać czasopisma poprzez prenumeratę dewizową (wpłata dokonywana poza granicami Polski w dewizach, wg cennika dewizowego z cenami podanymi w dolarach amerykańskich) lub poprzez zamówioną w kraju prenumeratę ze zleceniem wysyłki za granicę (zamawiający podaje dokładny adres odbiorcy za granicą, dokonując równocześnie wpłaty w wysokości dwukrotnie wyższej niż cena normalnej prenumeraty krajowej).

Egzemplarze archiwalne (sprzedaż przelewowa lub za zaliczeniem pocztowym) można zamawiać pisemnie, kierując zamówienia pod adresem: Wydawnictwo SIGMA NOT, Spółka z o.o. Zakład Kolportażu, 00-716 Warszawa, ul. Bartycka 20, paw. B, tel. 40-37-31, natomiast za gotówkę można je nabyć w Klubie Prasy Technicznej w Warszawie ul. Mazowieckiej 12, tel. 26-80-17.

Istnieje możliwość zaprenumerowania 1 egz. czasopisma po cenie ulgowej przez indywidualnych członków stowarzyszeń naukowo-technicznych zrzeszonych w FSNT oraz przez uczniów zawodowych i studentów szkół wyższych. Blankiet wpłaty na prenumeratę ulgową musi być opatrzony na wszystkich odcinkach pieczęcią koła SNT lub szkoły.

W przypadku zmiany cen w okresie objętym prenumeratą Wydawnictwo zastrzega sobie prawo do wystąpienia o dopłatę różnicy cen oraz prawo do realizowania prenumeraty tylko w pełni opłaconej.
Cena jednego egzemplarza: normalna 4,50 zł (45 000 zł), ulgowa 3,15 zł (31 500 zł)
Wartość prenumeraty w zł:
Normalna: kwartalna 13,50 zł (135 000 zł), półroczna 27 zł (270 000 zł), roczna 54 zł (540 000 zł)
Ulgowa: kwartalna 9,45 zł (94 500 zł), półroczna 18,90 zł (189 000 zł), roczna 37,80 zł (378 000 zł)

Skład i druk: Drukarnia SIGMA NOT Sp. z o.o. z. 245/95

P18 77/95

II Konferencja
Systemy Czasu Rzeczywistego
Wytwarzanie oprogramowania do zastosowań mających wpływ na bezpieczeństwo

Dynamiczny rozwój zastosowań nowych technologii informatycznych w coraz większym stopniu uzależnia człowieka od komputerów i wdrażanego w wielu zastosowaniach oprogramowania użytkowego (np. systemy sterowania ruchem kolejowym, komputerowe systemy sterowania lotem, systemy sterowania liniami produkcyjnymi pracującymi w systemie pracy ciągłej, systemy wspomagające wykonywanie skomplikowanych operacji medycznych, systemy gromadzenia i udostępniania informacji pracujących w czasie rzeczywistym itp.). Powstaje problem bezpieczeństwa oprogramowania. Jak należy tworzyć oprogramowanie systemów użytkowych, jakimi narzędziami i jakimi metodami, by spełniało ono normy jakościowe.

Problem jest bardzo ważny w przypadku pracy tych systemów w reżimie czasu rzeczywistego. Wadliwie skonstruowane oprogramowanie, wykrycie błędu w trakcie eksploatacji systemów – mogą doprowadzić do katastrof ekologicznych, tworzyć zagrożenia dla bezpieczeństwa życia ludzkiego, czy narażać na straty materialne użytkowników. Powstaje wiele nowych problemów – nawet natury organizacyjno-prawnej – np. związanych z ponoszeniem odpowiedzialności za wadliwe działanie systemów. W środowiskach naukowych, prowadzących badania w tej dziedzinie, istnieje wiele kontrowersji dotyczących metod i technologii, związanych z bezpieczeństwem. Jednym z podstawowych celów tegorocznej konferencji jest wymiana poglądów i doświadczeń pracowników z różnych ośrodków naukowo-badawczych zajmujących się tworzeniem bezpiecznego oprogramowania.

Pierwsza Ogólnopolska Konferencja Naukowo-Techniczna *Oprogramowanie Komputerowych Systemów Czasu Rzeczywistego,* która odbyła się we wrześniu 1994 roku we Wrocławiu, spotkała się z dużym zainteresowaniem. Uczestniczyli w niej liczni przedstawiciele środowisk uczelnianych i przemysłowych. Materialnym plonem Konferencji – poza wydanymi materiałami z wygłoszonymi referatami – był także cykl artykułów publikowanych na łamach *INFORMATYKI,* w ramach – powstałego z inicjatywy uczestników Konferencji – działu *Systemy Czasu Rzeczywistego.* W trakcie ubiegłorocznej Konferencji powstała myśl o zainicjowaniu regularnego cyklu konferencji, poświęconych systemom czasu rzeczywistego, i otwarcia konferencji także dla uczestników zagranicznych. Pomimo krótkiego czasu udało się utworzyć Komitet Programowy, złożony nie tylko z przedstawicieli większych krajowych ośrodków naukowych, ale także z przedstawicieli kilku innych krajów. Do współpracy przy organizacji konferencji pozyskano Komitet IEEE (*Industrial Electronic Society*). Współpraca ta stwarza realne perspektywy nadania kolejnym konferencjom szerokiego zasięgu międzynarodowego.

Tegoroczna II Konferencja *Systemy Czasu Rzeczywistego* jest ukierunkowana na zagadnienia związane z wytwarzaniem oprogramowania dla zastosowań, mających wpływ na bezpieczeństwo. Wybór tych zagadnień jako wiodących wynika z obserwacji, że większość istotnych zastosowań komputerowych systemów czasu rzeczywistego wiąże się ze sterowaniem urządzeniami lub systemami, od których oczekuje się bezawaryjnego funkcjonowania. Zakłócenia w działaniu takich systemów mogą powodować straty materialne, a nawet narażać ludzi na niebezpieczeństwo.

Nadesłane referaty zakwalifikowano do grup tematycznych:

- Bezpieczeństwo i niezawodność oprogramowania;
- Metody specyfikacji i projektowania;
- Metody i narzędzia implementacji;
- Normalizacja i ocena oprogramowania;
- Atestacja oprogramowania;
- Zastosowania.

Każdy nadesłany referat był recenzowany przez dwóch recenzentów, wybitnych specjalistów z poszczególnych grup tematycznych. Recenzje były bardzo wnikliwe. Recenzenci pochodzą z Australii, Francji, Japonii, USA i polskich akademickich ośrodków naukowych. Łącznie przyjęto 34 referaty z czterech krajów. Spośród nich wybrano do publikacji w specjalnym numerze *INFORMATYKI* te, które uznano za reprezentatywne dla zakresu i charakteru konferencji. Inne wybrane materiały będą publikowane w kolejnych numerach *INFORMATYKI.*

Materiały konferencyjne zostały także wydane przez Oficynę Wydawniczą Politechniki Wrocławskiej.

Tegoroczna konferencja ma charakter międzynarodowy, chociaż przeważająca liczba uczestników pochodzi z Polski. Z tego względu przyjęto język polski i angielski jako języki robocze, a ponadto – specjalnie dla uczestników krajowych, a zwłaszcza dla tych, którzy dopiero zaczynają zajmować się zagadnieniami czasu rzeczywistego – konferencję poprzedza cykl wykładów szkoleniowych. Niektóre z nich publikujemy w bieżącym numerze *INFORMATYKI.*

Prof. dr hab. Zygmunt Mazur
Przewodniczący Komitetu Organizacyjnego

Prof. dr hab. inż. Zbigniew Huzar
Przewodniczący Komitetu Programowego

publikacje

Problemy jakości w projektowaniu oprogramowania systemów wbudowanych

Krzysztof Sacha
Instytut Automatyki i Informatyki Stosowanej
Politechnika Warszawska

Celem wszystkich metod projektowania oprogramowania jest podniesienie jakości programów i obniżenie kosztów ich opracowania. Niestety, jakości programów nie można mierzyć żadnym pojedynczym wskaźnikiem. Co więcej, samo pojęcie jakości programów nie jest dostatecznie dobrze określone. Celem tego artykułu jest wskazanie kilku istotnych czynników jakości oprogramowania, i odniesienie do nich podstawowych metod projektowych.

Nieustanna dyskusja nad problemem jakości oprogramowania nie doprowadziła jeszcze do sprecyzowania żadnych ilościowych miar jakości. Jednak udało się wyodrębnić zespół kilku cech, którymi powinny odznaczać się „dobre" programy [4, 2]. Do najważniejszych czynników jakości można zaliczyć:

- poprawność (ang. *correctness*),
- odporność (ang. *robustness*),
- weryfikowalność (ang. *verifiability*),
- modyfikowalność (ang. *modifiability*),
- powtarzalność wykorzystania (ang. *reusability*).

Czynniki jakości oprogramowania

Poprawność definiuje się przeważnie jako zdolność programu do wypełniania funkcji, określonych w specyfikacji wymagań, lub jako zdolność do spełnienia oczekiwań użytkownika. Praktyczne znaczenie ma tylko druga z tych definicji, gdyż satysfakcja użytkownika jest jedyną miarą sukcesu każdego projektu programowego. W tradycyjnym ujęciu, poprawność określa się przez relację, zachodzącą między danymi wejściowymi, a wynikami wykonania programu. W dziedzinie systemów wbudowanych takie podejście jest nieadekwatne, gdyż wymagania użytkownika określają nie tyle wynik przetwarzania, ile pewne ciągłe zachowanie. Nieodłącznym elementem poprawności zachowania jest dotrzymanie narzuconego czasu reakcji systemu. Stosowane tu metody muszą pozwalać na specyfikowanie zachowań oraz określanie i weryfikację ograniczeń czasowych.

Odporność oznacza zdolność programu do akceptowalnego zachowania w warunkach nienormalnych, nie ujętych w specyfikacji wymagań (nieznanych). W dziedzinie systemów wbudowanych, w których nadrzędnym wymaganiem jest zapewnienie bezpieczeństwa instalacji, za „akceptowalne" uznaje się na ogół zachowania „bezpieczne". Poprawność i odporność decydują łącznie o pewności działania oprogramowania (ang. *dependability*). Z samej swojej istoty odporność jest cechą słabo określoną, ma jednak decydujące znaczenie dla zachowania bezpieczeństwa po wystąpieniu awarii lub przeciążenia systemu.

Warto zauważyć różnicę między pojęciami odporności, a niezawodności. Wymaganie niezawodności oznacza konieczność stałej poprawnej pracy, i w pewnym sensie jest ostrzejsze od wymagania odporności. Z drugiej strony, absolutna niezawodność jest niemożliwa do osiągnięcia, i wymagania niezawodnościowe charakteryzuje się na ogół statystycznie, np. przez prawdopodobieństwo błędu lub awarii systemu. Takie podejście jest nie do przyjęcia w odpowiedzialnych zastosowaniach sterujących, w których użytkownik może zgodzić się na błędy i awarie, natomiast nie może zgodzić się na powstanie sytuacji niebezpiecznej.

Weryfikowalność oznacza łatwość sprawdzenia poprawności programu. W każdym wypadku cecha ta ma ogromne znaczenie ekonomiczne, gdyż koszty weryfikacji sięgają 50% kosztów typowego projektu [13]. W dziedzi-

nie systemów wbudowanych znaczenie tej cechy jest szczególnie duże, gdyż katastrofalne skutki ewentualnych błędów i awarii wymuszają szczególnie staranną i dokładną weryfikację programów.

Modyfikowalność oznacza łatwość adaptacji oprogramowania do zmienionych wymagań lub do zmienionego środowiska wykonawczego. Cecha ta ma ogromne znaczenie ekonomiczne, gdyż oprogramowanie wbudowane jest na ogół eksploatowane wiele lat, i podlega w tym czasie wielu zmianom, wymuszanym przez okresowe modernizacje instalacji sterowanej. Łączne koszty zmian i modyfikacji eksploatowanego oprogramowania przekraczają na ogół koszty opracowania programów i sięgają 200% kosztów oryginalnego projektu [13]. Weryfikowalność i modyfikowalność określają łącznie łatwość konserwacji programów (ang. *maintainability*).

Powtarzalność wykorzystania oznacza łatwość wielokrotnego wykorzystania tych samych programów, lub ich komponentów, w różnych projektach i różnych zastosowaniach. Cecha ta wpływa dodatnio na wszystkie inne wskaźniki jakości. Wielokrotne wykorzystanie poprawnych i odpornych komponentów sprzyja przyspieszeniu oryginalnego projektu, i zmniejszeniu jego kosztów, ułatwia osiągnięcie poprawności nowych programów oraz zmniejsza koszty weryfikacji i konserwacji.

Inne wskaźniki jakości oprogramowania rozważane w literaturze [4, 2], a nie uwzględnione w tej pracy, obejmują:

- Otwartość (ang. *compatibility*), czyli łatwość współpracy oprogramowania z innymi systemami. Cecha ta nie jest bezpośrednio związana z zastosowaniem takich, a nie innych metod projektowania, lecz z podjęcia określonych decyzji projektowych; otwartość programu wynika na ogół z wykorzystania standardowych metod organizacji plików i baz danych, i standardowych protokołów komunikacyjnych.
- Efektywność (ang. *efficiency*), czyli skuteczność i oszczędność wykorzystania zasobów systemu. Ta niewątpliwie ważna cecha programu jest związana bardziej z fazą pakietyzacji i implementacji, niż projektowania. W miarę spadku kosztów sprzętu znaczenie efektywnego wykorzystania zasobów jest coraz mniejsze.
- Przenośność (ang. *portability*), czyli łatwość przenoszenia oprogramowania do innych środowisk wykonawczych (inne komputery, systemy operacyjne). Cecha związana raczej z fazą implementacji, a ściślej, z rodzajem wykorzystywanego środowiska programowego. Na przykład, programy opracowane w języku Ada można z definicji instalować w dowolnym systemie wyposażonym w kompilator tego języka.
- Szczelność (ang. *integrity*), czyli zdolność ochrony przed nieupoważnionym dostępem. Moim zdaniem cecha ta nie stanowi samodzielnej zalety lub wady oprogramowania, jeżeli jest istotna w danym zastosowaniu, to powinna znaleźć wyraz w specyfikacji wymagań.
- Łatwość użycia (ang. *ease of use*), mimo niewątpliwej ważności jest to cecha zbyt subiektywna i trudno mierzalna dla ujęcia w jakichkolwiek ścisłych rozważaniach.

Problemy rozwoju oprogramowania

Historycznie najstarszy sposób opracowania programów stanowił jednofazowy proces, w którym programista tworzył program w sposób intuicyjny, bez systematycznych działań projektowych i systematycznej weryfikacji. W miarę wzrostu stopnia złożoności problemów rozwiązywanych za pomocą komputerów, widoczna stała się potrzeba rozdzielenia etapów:

- analizy wymagań, prowadzącej do specyfikacji wymagań użytkownika,
- projektowania, prowadzącego do określenia struktury programu,
- implementacji,
- weryfikacji, realizowanej przede wszystkim przez testowanie.

Kolejne etapy procesu rozwoju oprogramowania są wykonywane sekwencyjnie, jeden po drugim, a ich wynikami są określone dokumenty projektowe. Opracowanie i zatwierdzenie końcowych dokumentów etapu jest warunkiem rozpoczęcia następnego etapu pracy.

Tak zorganizowany proces projektowy określa kolejność podejmowania coraz bardziej szczegółowych decyzji projektowych i umożliwia systematyczne prowadzenie pracy. W obecnym stanie inżynierii oprogramowania jest to dominujący sposób prowadzenia prac programowych w naszym kraju, w tych ośrodkach, w których w ogóle wdraża się jakiś sposób zarządzania pracami programowymi.

Doświadczenie praktyczne pokazało jednak, że proces projektowy oparty wyłącznie na dokumentach tekstowych, bez jakiegoś zaplecza matematycznego, nie może zagwarantować powodzenia [20, 1]. Celem tego rozdziału jest przedstawienie potencjalnych problemów i niedostatków takiego procesu, których odkrycie stanowiło bodziec dla podjęcia prac nad różnymi metodami projektowania programów.

Pierwszym etapem rozwoju oprogramowania jest analiza wymagań użytkownika. Wynikiem analizy jest dokument, nazywany *specyfikacją wymagań* (lub *założeniami na oprogramowanie*), który opisuje w języku naturalnym sposób działania projektowanego programu. Specyfikację – która ma zazwyczaj rozmiary sporej książki – przedstawia się do oceny i akceptacji przyszłemu użytkownikowi, a po zatwierdzeniu przyjmuje za wzorzec dla projektowania i oceny programów.

Słabą stroną tej procedury jest niska wiarygodność oceny specyfikacji dokonanej przez użytkownika. Składają się na to dwie przyczyny:

- niezdolność użytkownika do dokładnego sprecyzowania wymagań stawianych systemowi komputerowemu;
- niejednoznaczność języka naturalnego, prowadząca nieuchronnie do niejednoznaczności specyfikacji.

W rezultacie, specyfikacja wymagań nie odzwierciedla często rzeczywistych potrzeb użytkownika, co prowadzi do powstania niepoprawnego oprogramowania, które musi być natychmiast modyfikowane.

W etapach projektu i implementacji następuje określenie i optymalizacja struktury oprogramowania. Sposób optymalizacji zależy od przyjętego kryterium jakości, którym może być np. czas wykonania, zajętość pamięci lub przenośność i łatwość modyfikacji. Etapy projektu i implementacji są wykonywane ręcznie, a ich wykonanie wymaga twórczej pracy projektantów i programistów.

Weryfikacja zgodności implementacji ze specyfikacją wymagań następuje podczas testowania programu.

Słabą stroną tej procedury jest fakt, że testowaniu mogą podlegać jedynie gotowe programy. Uniemożliwia to bieżące weryfikowanie kolejnych decyzji projektowych i szybką korekcję błędów. Poprawienie błędów projektu po zakończeniu implementacji jest zazwyczaj bardzo kosztowne, gdyż wymaga powtórzenia części projektu, programowania i testowania. Proces testowania nie może udowodnić poprawności programu.

W fazie konserwacji kolejne modyfikacje wykonuje się z reguły wprost na tekstach programów. Zmiany wprowadza się lokalnie, bez przeprowadzania globalnej optymalizacji całego programu. Pogarsza to coraz bardziej strukturę oprogramowania i czyni kolejne modyfikacje coraz trudniejszymi. Co więcej, modyfikacji programów nie zawsze towarzyszy pełna aktualizacja dokumentacji projektowej, tzn. specyfikacji wymagań i opisu projektu, co dodatkowo utrudnia i powiększa koszty dalszej konserwacji.

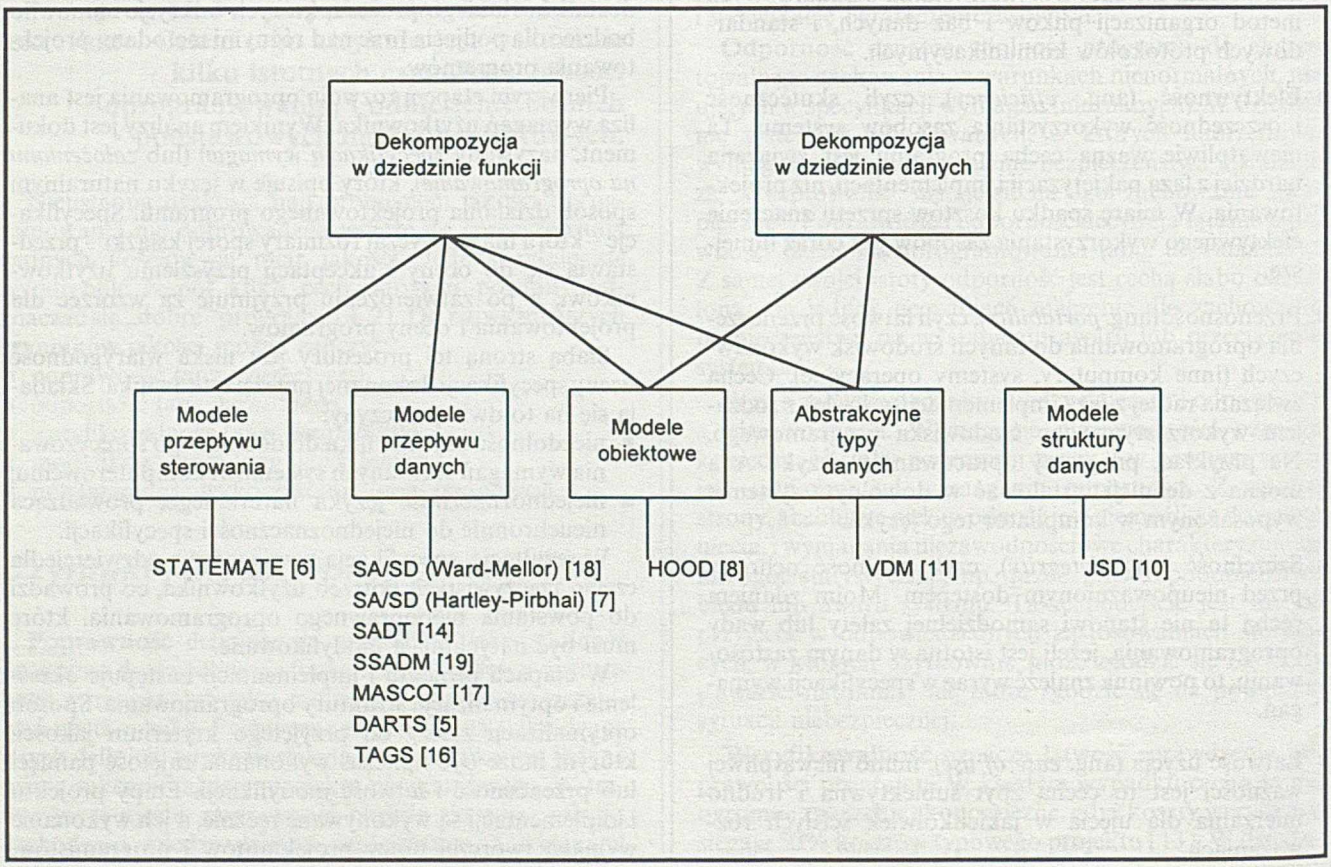
Podstawowe problemy przedstawionego podejścia do projektowania oprogramowania można podsumować następująco:

- Brak jednoznacznego języka komunikacji analityków, opracowujących specyfikację wymagań, z użytkownikami, prowadzący do niedokładnego określenia potrzeb użytkownika.
- Niejednoznaczność specyfikacji wymagań, wprowadzona przez naturalny język tego dokumentu.
- Niski stopień automatyzacji w podstawowych etapach pracy: analizie wymagań, projektowaniu, programowaniu i testowaniu.
- Niska wiarygodność i bardzo wysoki koszt weryfikacji programów, przeprowadzanej przez testowanie.
- Bardzo wysoki koszt konserwacji, która systematycznie rujnuje strukturę programów i utrudnia wprowadzanie dalszych zmian.

Klasyfikacja metod projektowania

Celem procesu projektowego jest opracowanie struktury programu, tzn. dokonanie podziału na jednostki programowe, określenie sprzęgów między jednostkami oraz zdefiniowanie podstawowych struktur danych i algorytmów wykonywanych przez poszczególne jednostki. Kluczowy punkt tego procesu – określenie podziału na jednostki programowe – rozpoczyna się na ogół od budowy pewngo modelu przetwarzania, którego celem jest dekompozycja problemu na mniejsze, łatwiejsze do opanowania elementy. Sposób podziału i rodzaj modelu determinują podstawowe cechy struktury programu powstającego przy zastosowaniu danej metody, i co za tym idzie, wywierają decydujący wpływ na jakość powstającego oprogramowania.

W każdym problemie przetwarzania danych występują zawsze dwa składniki: dane, opisujące dziedzinę problemu i istniejące w niej zależności, oraz funkcje, opisujące wymagane przetwarzanie. W końcowym wyniku dekompozycja obejmuje zawsze obydwa składniki, ale sposób dochodzenia do ostatecznej struktury programu może prowadzić przez analizę wymaganych funkcji albo przez analizę występujących w dziedzinie problemu danych. Zależnie od dokonanego wyboru otrzymuje się dwie zasadnicze drogi podziału:



Rys. 1. Klasyfikacja metod projektowania systemów wbudowanych

- podział w dziedzinie funkcji,
- podział w dziedzinie danych.

W ramach tych dwóch dróg, oraz na granicy między nimi, rozwinęło się wiele metodologii projektowania, opierających dekompozycję na różnych podstawach matematycznych. Można tu wyróżnić pięć podstawowych modeli dekompozycji (rys. 1):

- przepływu sterowania,
- przepływu danych,
- abstrakcyjnych typów danych,
- struktury obiektów,
- struktury danych.

Modele dekompozycji określają pewne ogólne kryteria, w ramach których powstały szczegółowe metody, definiujące sposób postępowania i notację. Większość praktycznie użytecznych metod nie ogranicza się do pojedynczego modelu dekompozycji, lecz korzysta z kilku modeli, używanych do opisu różnych aspektów problemu. Przedstawiona klasyfikacja uwzględnia tylko podstawowy model dekompozycji.

Metody strukturalne, oparte na modelu przepływu danych, zajmują obecnie dominującą pozycję na rynku. Wniosek ten wypływa natychmiast z porównania liczby metod wymienionych na rys. 1 w różnych grupach klasyfikacyjnych. Metody strukturalne oferują sprawdzoną technologię specyfikacji i projektowania, oraz liczne, komercyjnie dostępne, systemy wspomagające. Zalety tych metod uwidaczniają się najpełniej w wielkich projektach programowych, w których decydujące znaczenie ma organizacja i dyscyplina prowadzenia prac projektowych. Podstawowe zarzuty, stawiane metodom strukturalnym, dotyczą niskiej modyfikowalności i powtarzalności wykorzystania otrzymywanych programów.

Metody obiektowe stają się coraz wyraźniej rynkową alternatywą metod strukturalnych, zyskując popularność w małych i średnich projektach, zwłaszcza zorientowanych na implementację w językach Ada i C++. Dzięki odmiennemu modelowi dekompozycji, metody te prowadzą do bardzo stabilnej struktury programu, bardzo podatnej na modyfikacje i wielokrotne wykorzystanie komponentów. Niebywale wysokie koszty konserwacji i rosnący nacisk na powtarzalne wykorzystanie programów są jednym z głównych powodów wzrostu popularności metod obiektowych. Mimo swoich zalet, metody te nie osiągnęły jeszcze pełnej dojrzałości, i nie zostały zweryfikowane w dużych projektach programowych.

Metody oparte na modelu struktur danych zostały opracowane przede wszystkim dla projektowania systemów przetwarzania danych ekonomicznych i administracyjnych. Obecnie wydaje się, że ta gałąź klasyfikacji zakończy się ślepo, gdyż dekompozycja w dziedzinie danych wyewoluowała w stronę znacznie bardziej uniwersalnych metod obiektowych. Duże znaczenie zachowują jedynie diagramy powiązań danych (ang. *ERD – entity-relationship diagram*), które stały się uznanym narzędziem specyfikowania danych, zaadaptowanym przez wiele różnych metod. Diagramy te odgrywają szczególnie dużą rolę w swym naturalnym obszarze zastosowania – specyfikacji baz danych.

Metody oparte na modelach przepływu sterowania są stosowane przede wszystkim do specyfikacji oprogramowania systemów sterowania dowodzenia. Większość tych metod nie wyszła poza stadium badań eksperymentalnych. Wyjątkiem jest metoda *STATEMATE*, związana z systemem wspomagającym o tej samej nazwie, która jest wykorzystywana do projektowania oprogramowania awionicznego przez wojskowy przemysł Izraela.

Metody formalne, oparte na teorii abstrakcyjnych typów danych, stwarzają nadzieję na opanowanie metod dowodzenia poprawności programów i tym samym wyeliminowanie testowania. Jak na razie, jedyną komercyjnie dostępną metodą formalnego konstruowania programów jest metoda VDM, która jednak nie umożliwia określania ciągłych zachowań oprogramowania i specyfikowania wymagań czasowych. Braki te uniemożliwiają wykorzystanie metody VDM do projektowania oprogramowania systemów wbudowanych. Mimo krytycznej oceny przydatności metod formalnych w obecnym stanie ich rozwoju trzeba podkreślić, że znaczenie tych metod powoli, lecz nieustannie rośnie. Szczególnym optymizmem napawają tu udane próby włączenia metod opartych na teorii kolorowych sieci Petriego do weryfikacji zachowań oprogamowania projektowanego metodą SADT [9].

Modelowanie przepływu danych

Grafowe modele przepływu danych stanowią jeden z najstarszych sposobów przedstawiania obiegu informacji, stosowany tradycyjnie do opisu systemów administracyjnych, bankowych i komercyjnych. Wykorzystanie tych metod do modelowania oprogramowania doprowadziło w latach siedemdziesiątych do powstania dużej grupy metod, obejmowanych zbiorczą nazwą analizy i projektowania strukturalnego. Wszystkie metody tej grupy opierają się na zstępującym podejściu do projektowania oprogramowania i są wkomponowane w proces projektowy oparty na modelu kaskadowym.

Grafy przepływu danych bardzo dobrze nadają się do opisu struktury przetwarzania danych, nie dają natomiast możliwości precyzyjnego wyrażenia wymagań narzuconych na przepływ sterowania, a zwłaszcza wymagań czasowych. Stąd, metody przeznaczone do projektowania oprogramowania systemów wbudowanych wykorzystują pomocnicze modele przepływu sterowania, w tym zwłaszcza modele automatowe.

Niezależnie od różnic występujących między różnymi metodami strukturalnymi, generalna linia postępowania jest zawsze podobna, i polega na stopniowym podziale problemu na coraz bardziej szczegółowe fragmenty, zgodnie z następującym schematem:

1. Określ granice oprogramowania i środowiska, w którym to oprogramowanie działa, oraz zbadaj zdarzenia, na które system musi odpowiadać.
2. Zbuduj model przetwarzania zdarzeń i danych wejściowych, w formie grafu przepływu danych.
3. Podziel graf między współbieżne zadania i uzupełnij szczegóły związane z komunikacją zadań z innymi zadaniami i ze światem zewnętrznym.
4. Dla każdego zadania: przekształć strukturę grafu przepływu danych w hierarchię wywołujących się jednostek programowych.
5. Zoptymalizuj strukturę programu łącząc, bądź dzieląc, jednostki programowe dla zwiększenia wydajności i szybkości działania.

Dla ilustracji metody rozważmy oprogramowanie sterujące instalacją złożoną ze zbiornika, czujnika poziomu, zaworu dopływowego, zaworu odpływowego i drukarki alarmów. Zadaniem programu jest stabilizacja poziomu cieczy w zbiorniku przez sterowanie położeniem zaworu

dopływowego; zawór odpływowy znajduje się poza kontrolą programu. Dodatkowe zadania programu obejmują filtrację pomiarów, ograniczenie wartości sygnału sterującego zaworem i kontrolę utrzymania poziomu w zadanych granicach. W razie przekroczenia poziomu granicznego, program drukuje komunikat alarmowy na drukarce.

Diagram (gryf) przepływu danych, opisujący przetwarzanie związane ze sterowaniem instalacją, przedstawia rys. 2.

Zgodnie z notacją używaną w metodzie Warda-Mellora [18], elementami diagramu są okręgi, reprezentujące transformacje danych (elementy czynne), otwarte prostokąty, reprezentujące zbiory lub struktury danych (elementy bierne), oraz skierowane łuki, reprezentujące przepływy danych lub sterowania. Łuki zakończone podwójnymi strzałkami oznaczają przepływy ciągłe, łuki zakończone pojedynczymi strzałkami oznaczają przepływy dyskretne (wiadomości), a łuki rysowane linią przerywaną symbolizują przepływy sterujące (dwustanowe). Opracowanie diagramu przepływu danych jest wynikiem analizy wymagań użytkownika, wykonanej w dwóch pierwszych krokach metody.

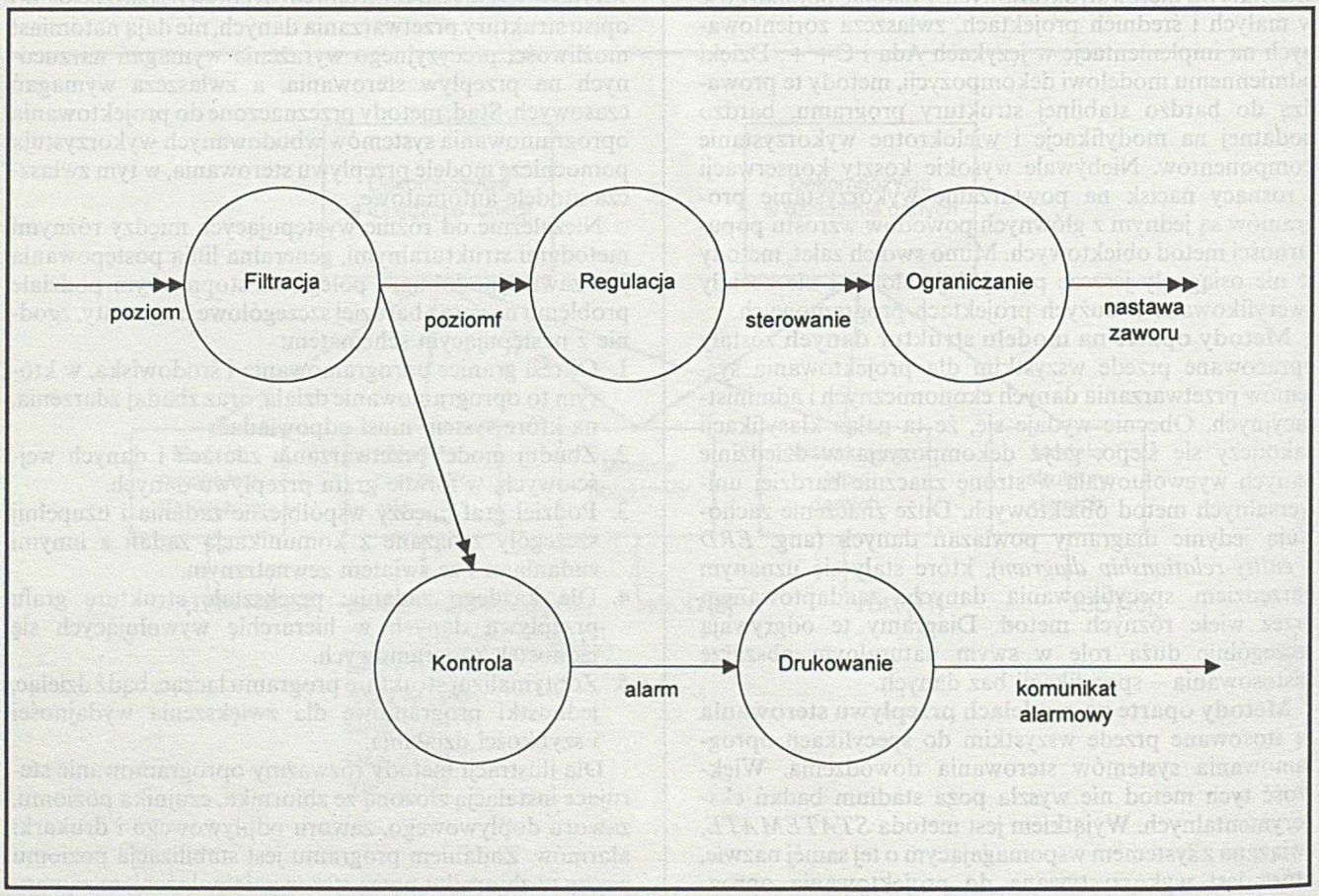
Trzeba podkreślić, że tylko wyjątkowo prosty system można przedstawić na pojedynczym diagramie przepływu danych. Opis przetwarzania w każdym nietrywialnym systemie wymaga sporządzenia hierarchii diagramów, w której transformacje występujące na digramach wyższego poziomu są rozwijane (dekomponowane) na diagramach niższego poziomu. Dekompozycja transformacji i rozwijanie hierarchii diagramów jest kontynuowane aż do poziomu, na którym wszystkie transformacje są dostatecznie proste, aby można je opisać np. zdaniami pseudokodu lub formułami matematycznymi (równaniami albo funkcjami).

Etap projektu przekształca funkcjonalny model przetwarzania w implementacyjny model programu. Celem tego etapu jest uzupełnienie modelu funkcjonalnego szczegółami dotyczącymi technologii realizacyjnej, i przekształcenie jego struktury w taką strukturę programu, która pozwoli spełnić ograniczenia zasobowe i osiągnąć wymaganą wydajność. Najważniejsze decyzje projektowe, podejmowane w tym etapie, określają podział programów na współbieżne zadania, sposób uruchamiania i komunikowania się zadań oraz podział programów zadań na podprogramy, a także algorytmy podprogramów i podstawowe struktury danych. Problemy te nie były brane pod uwagę podczas analizy wymagań; podczas projektu stanowią ważne problemy decyzyjne, w których różne odpowiedzi prowadzą do różnych implementacji.

Wróćmy do przykładu systemu sterującego poziomem cieczy w zbiorniku. Pytania, które trzeba postawić podczas projektowania programu, to np.:

- Czy pomiary, sterowanie i wydruki alarmów ma realizować jeden program sekwencyjny, czy zespół zadań?
- Jeżeli oprogramowanie będzie składać się z wielu zadań, to jak będzie wyglądać komunikacja tych zadań?
- W jaki sposób mają być odczytywane wskazania czujnika poziomu?

Rys. 2. Model funkcjonalny oprogramowania sterującego



Wykonanie trzeciego kroku metody sprowadza się w tym przykładzie do zastąpienia transformacji ciągłych, tzn. produkujących sygnały ciągłe, transformacjami dyskretnymi, wykonywanymi cyklicznie w ustalonym rytmie czasowym, oraz do podziału diagramu między dwa zadania: cykliczne zadanie pomiarów i sterowania, oraz zadanie drukowania komunikatów alarmowych. Rozdzielenie zadań jest konieczne, gdyż czas drukowania komunikatu jest dłuższy od wymaganego okresu powtarzania pomiarów. Komunikacja zadań wymaga wprowadzenia zbioru pośredniczącego (bufora), przechowującego przeznaczone do wydrukowania komunikaty o przekroczeniu poziomu alarmowego, lub o powrocie poziomu do normy. Diagram przepływu danych zadania pomiarów i sterowania przedstawia rys. 3.

Czwarty krok metody – przekształcenie grafu przepływu danych w hierarchię wywołań podprogramów – rozpoczyna się od wybrania „transformacji centralnej", która stanie się ośrodkiem przekształcenia. Transformacja centralna, gałęzie grafu dochodzące do niej i gałęzie grafu wychodzące z niej stają się podprogramami najwyższego poziomu, wywoływanymi bezpośrednio z programu głównego. Rozwinięcie dalszych poziomów hierarchii następuje w wyniku przekształcenia „gałęzi dochodzących" grafu i „gałęzi wychodzących". Proces przekształcenia przebiega w obydwu gałęziach nieco inaczej, zawsze jednak podprogramy odpowiadają transformacjom i przepływom między transformacjami gałęzi grafu. Hierarchiczna struktura programu sterującego poziomem cieczy w zbiorniku jest przedstawiona na rys. 4.

Optymalizacja struktury programu polega na ogół na łączeniu niektórych podprogramów lub przenoszeniu ich między poziomami hierarchii, tak aby uprościć definicje sprzęgów między podprogramami i przyspieszyć wykonanie całości. Proces ten zależy od szczegółowych cech problemu i środowiska wykonawczego i nie będzie dalej opisywany.

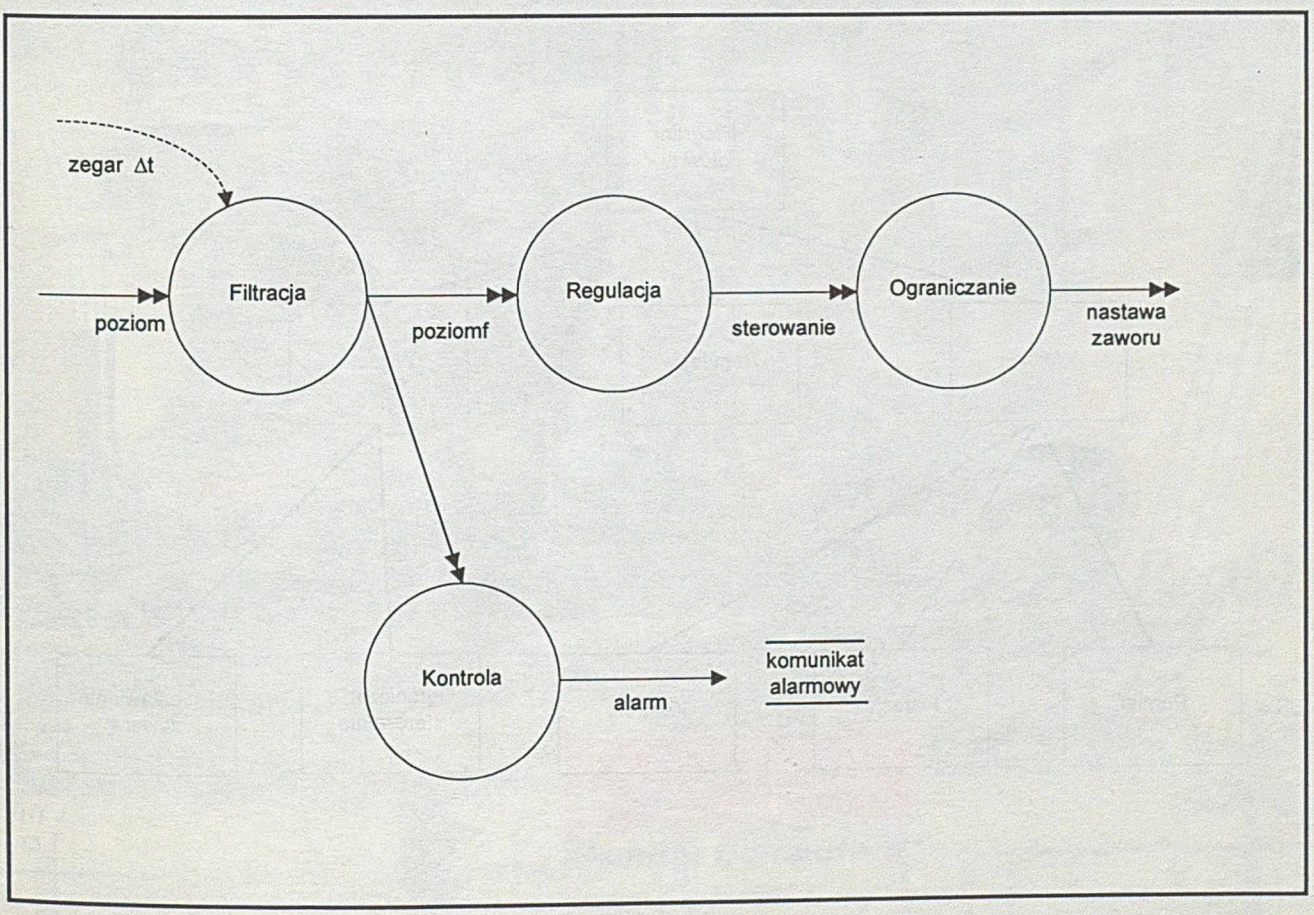
Na zakończenie rozważmy fazę konserwacji oprogramowania, i przypuśćmy, że konieczne jest dodanie do przykładowego systemu sterującego dodatkowej funkcji wyświetlania aktualnego poziomu cieczy na animowanym na bieżąco wykresie. Ponieważ czas rysowania wykresu jest dłuższy od żądanego czasu powtarzania pomiarów, rozszerzenie systemu będzie wymagało dodania nowego zadania rysującego. Nowe zadanie, biegnące w innym rytmie czasowym, musi otrzymywać dane od zadania pomiarów i sterowania, co zmusza do wprowadzenia znacznych zmian do już istniejącego kodu programu. To z kolei prowadzi do konieczności ponownego testowania niemal całego programu.

Ocena

Zaletą metod strukturalnych jest zorganizowana dyscyplina prowadzenia projektu, w której działanie rozpoczyna się od szczegółowego określenia wymagań, a każdy krok podejmuje się w kontekście określonym w kroku poprzednim. Umożliwia to bieżącą weryfikację poprawności kolejnych kroków projektu względem ustaleń poczynionych w krokach poprzednich – weryfikacja poprawności jest tu integralnie wbudowana w przebieg procesu projektowego.

Wady tego podejścia są wielorakie. Po pierwsze, nie ma ciągłości modelu i związanej z nim notacji na przestrzeni

Rys. 3. Model zadania pomiarów i sterowania



całego projektu. Określenie struktury programu wymaga gwałtownej zmiany rodzaju modelu i przejścia od grafu przepływu danych do kraty wywołań podprogramów. W nietrywialnych projektach przejście to jest trudne do przeprowadzenia, a co więcej, prowadzi do zatarcia związku między koncepcyjnym modelem przetwarzania a rzeczywistą strukturą programu. Po drugie, zasadnicze decyzje projektowe dotyczące podziału oprogramowania między komputery systemu, i między współbieżne zadania w każdym komputerze, są podejmowane stosunkowo wcześnie, kiedy nie jest jeszcze znana szczegółowa postać algorytmów. Nie daje to gwarancji spełnienia wymagań narzuconych na szybkość i wydajność przetwarzania i zmusza do przeprowadzenia optymalizacji, która nieuchronnie pogarsza logiczną strukturę projektu. To z kolei prowadzi do znacznego utrudnienia konserwacji. Trzecia wada wynika z faktu sztywnego usytuowania każdej jednostki projektowej w ograniczonym kontekście specyfikacji, określonej w poprzednim kroku projektu. Efektem takiego postępowania jest mała podatność na modyfikacje i trudność powtarzalnego wykorzystania fragmentów projektu w kontekście innych projektów.

Modelowanie struktury obiektowej

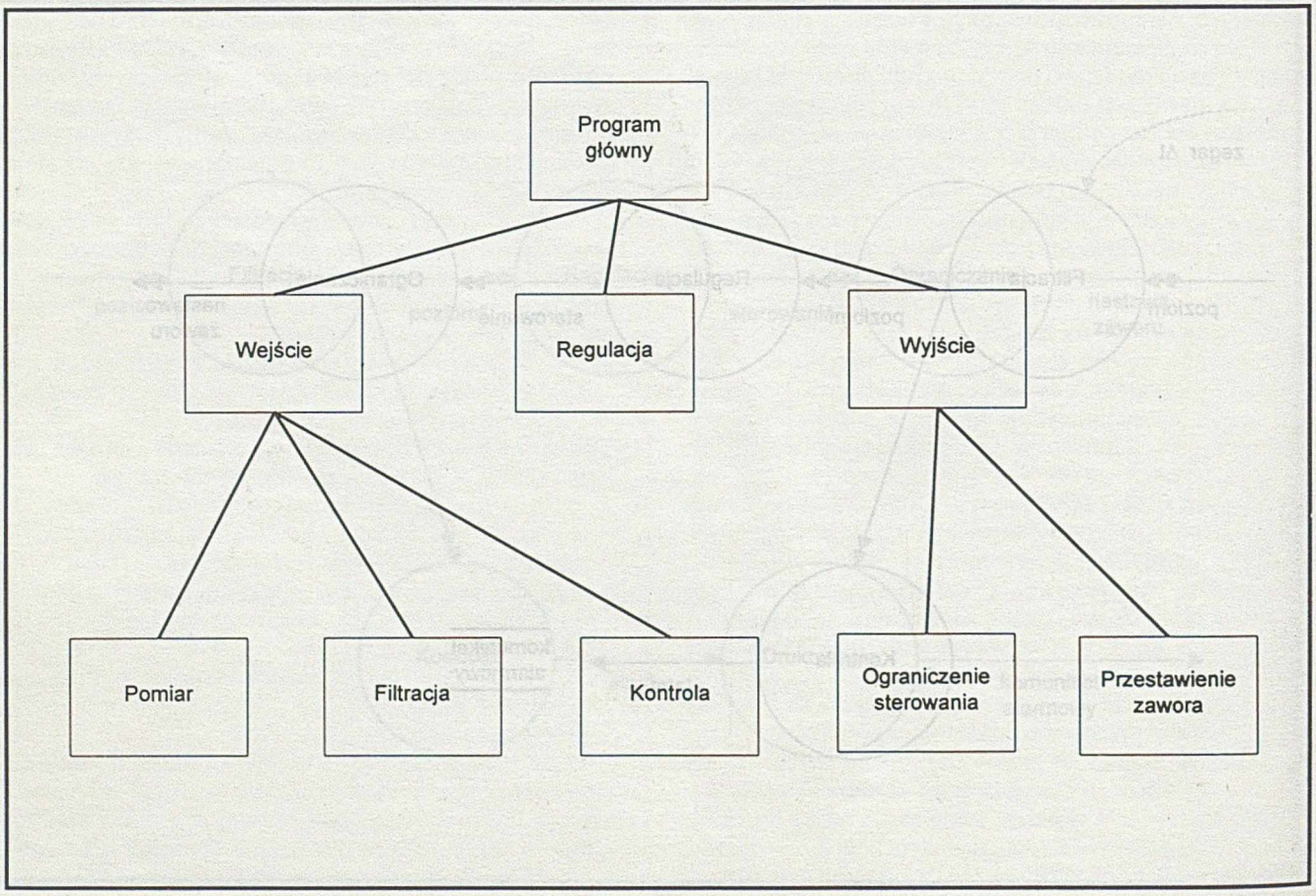
Naturalną reprezentację niemal każdej dziedziny zastosowania stanowi zbiór obiektów, występujących w realnym świecie, uzupełniony regułami współdziałania tych obiektów. Na przykład, w systemie graficznego sprzęgu komputera z użytkownikiem występują obiekty takie, jak: okienka, okienka dokumentów, okienka menu, paski przewijania, myszka itd. Obiekty są połączone wielorakimi zależnościami, np. pasek przewijania *jest częścią* okienka dokumentu, a okienko dokumentu *jest szczególnym przypadkiem* okienka.

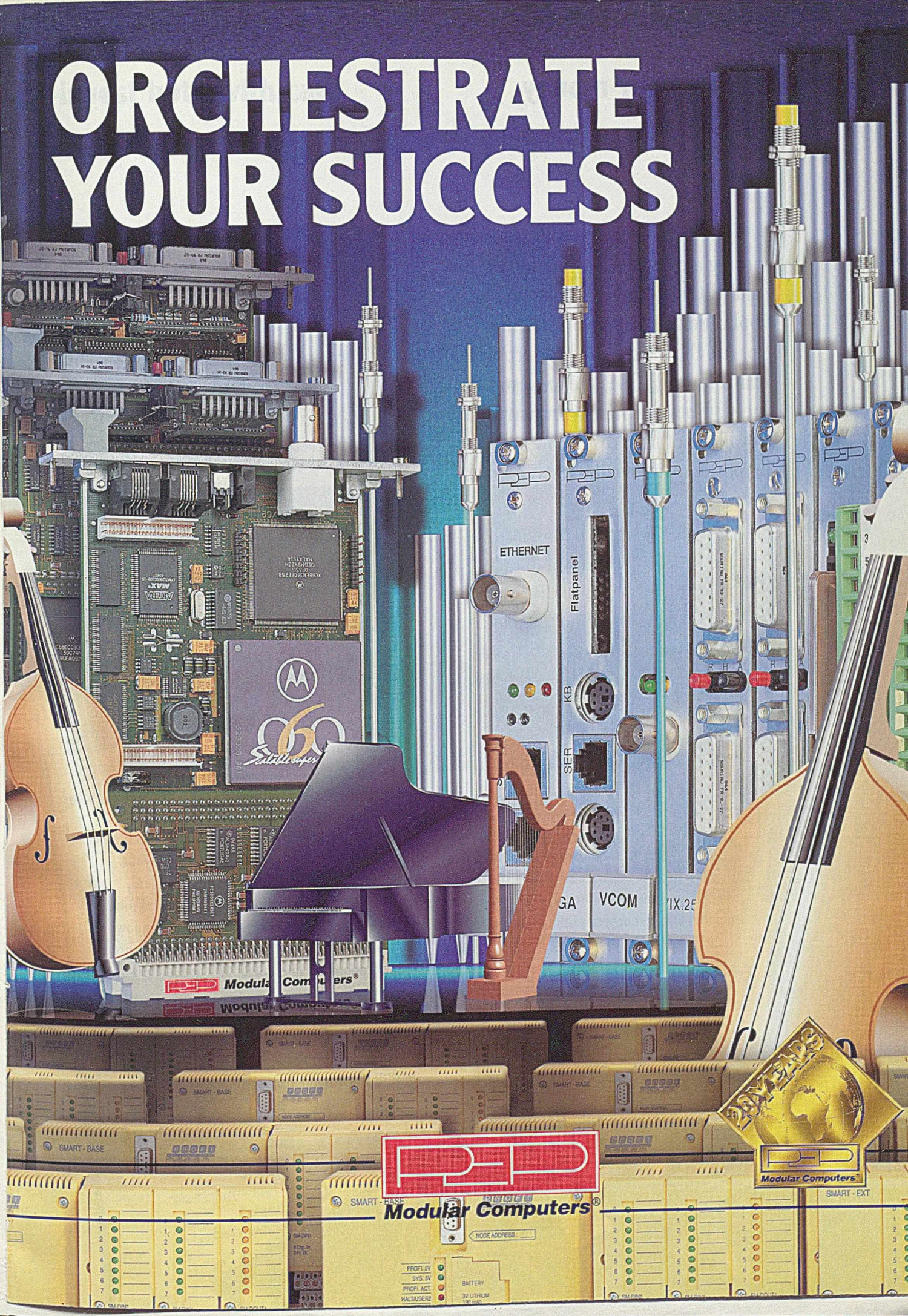
Metody obiektowe odwzorowują strukturę obiektów i ich połączeń w rzeczywistym świecie na strukturę obiektów programowych. Każdy obiekt programowy jest opisany przez atrybuty (dane) i usługi (operacje) na tych danych[1]. Na przykład, atrybutami okienka dokumentu mogą być: położenie, rozmiar, tekst zapisany wewnątrz okienka itd., a usługami: otwarcie, zamknięcie, przesunięcie, przewinięcie itd. Atrybuty i usługi stanowią nierozerwalną całość, w tym sensie, że usługi obiektu odnoszą się tylko do jego atrybutów, a wartości atrybutów obiektu mogą być zmieniane tylko przez jego usługi. Reguły działania całego systemu wyznacza wzajemne oddziaływanie obiektów. Na przykład, kliknięcie obiektu „myszka" na pasku przewijania wywołuje usługę przewinięcia obiektu „okienko dokumentu". Wywołanie usług jest jedynym sposobem komunikowania się obiektów.

Modele obiektowe bardzo dobrze nadają się do opisu zależności występujących w dziedzinie zastosowania. Relacja *jest częścią* umożliwia dekompozycję opisu całości na elementy prostsze, a relacja *jest szczególnym przypadkiem* pozwala na uproszczenie techniki opisu: obiekty stanowiące szczególne przypadki mogą *dziedziczyć opis* obiektu ogólniejszego i uzupełniać ten opis dodatkowymi specyficznymi atrybutami i usługami. Zmiany zachowania obiektu w czasie można opisywać za pomocą modeli

[1] Niektóre języki obiektowe nazywają usługi „metodami".

Rys. 4. Model implementacyjny zadania pomiarów i sterowania

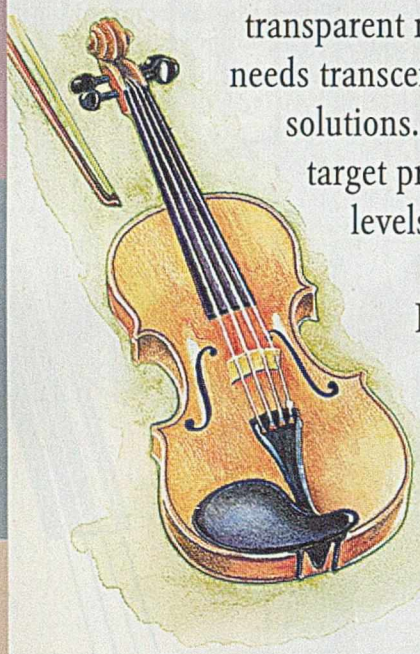



ORCHESTRATE
YOUR SUCCESS

Your time, your money

...orchestrate your success! Choosing a computer partner for VME and IEC1131 industrial automation applications places you in the role of Maestro. Are the instruments correct? The musicians trained? Does the orchestra fit the need? Does the price represent high value, not high cost? How will the composition survive the tests of time?

In the era of open systems and transparent networking, your needs transcend single vendor solutions. You must match target price/performance levels with expansion needs over time. Real-time system behavior is now a requirement.



When a global vendor earns the right to be your partner for these objectives, the resulting music becomes a legacy.

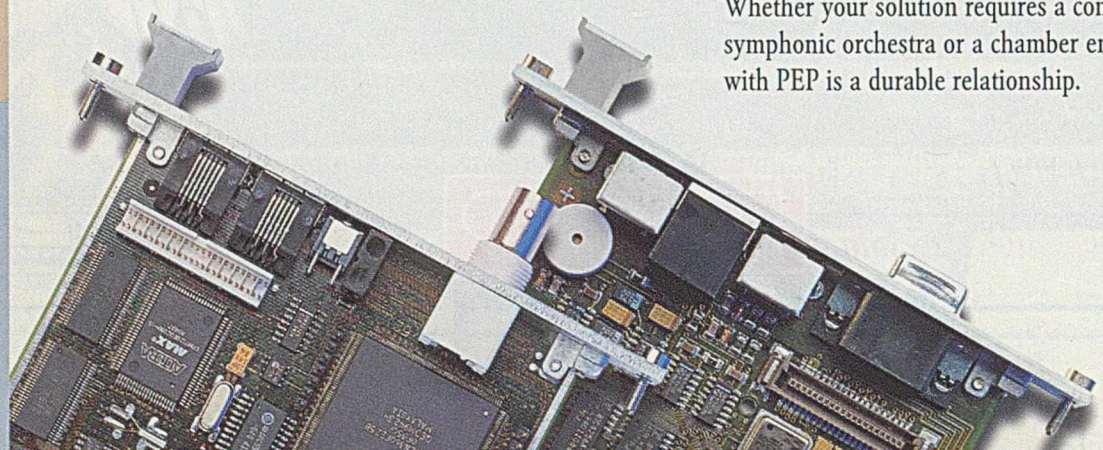
PEP Modular Computers is such a partner. A 20-year repertoire of open, interoperable solutions. A vast selection of high- performance, value-based modular and scalable hardware and software. Excellent products built to ISO9001 requirements. Skilled people. Technology partnerships with the best players. An impressive audience of satisfied customers (always front row center). Keen time-to-market sensitivity.

This symphony is well prepared!

You are not alone when PEP is your partner. The past foreshadows the future. Maestro, orchestrate your success. PEP helps you design complete solutions for your applications. Be the best through open, standard, interoperable implementations. Stay the best through key industry standards like VME, IEC1131-3, Windows , TCP/IP, PROFIBUS and more. It means excellence in standardization and interoperability.

A value-driven experience

Whether your solution requires a computer like a symphonic orchestra or a chamber ensemble, partnering with PEP is a durable relationship.



invest both with care

Computing. Will it ever be easy? So many decisions, so little time. You'd like to achieve a symphony but the sour notes of discordant architectures block the ear.

Perhaps you're trapped by single-vendor systems. Or out of room to grow. Worse, the de facto standard hardware can't take the heat... the cold... the vibration. But it was cheap. And you get to play games. Remember when you could never get in trouble by specifying a single famous vendor's proprietary solution! If it can't interoperate, you're in trouble.

«Open, standard, interoperable» sounds good, right? Add modular. smaller. faster. Compatible into the distant future. Sounds much better! Like a live concert.

Let's play what-if. What if a single vendor's warehouse door opened onto a vast selection, included a skilled docent as tour guide and guaranteed non-obsolescence. What do you call such a place? PEP.

International PEP Symphonies

Germany
Telecontrol, energy management

USA
Wafer manufacturing machines

Canada
Shutdown system nuclear power plants

France
Inflight equipment for helicopter, aircrafts

UK
Off-shore, oil rig safety systems

Benelux
Textile printing, Elevator control, Laser cutting machines

Scandinavia
Navy vessels

Korea
Crane control

Taiwan
Siren control for civil defence

Poland
Subway control system

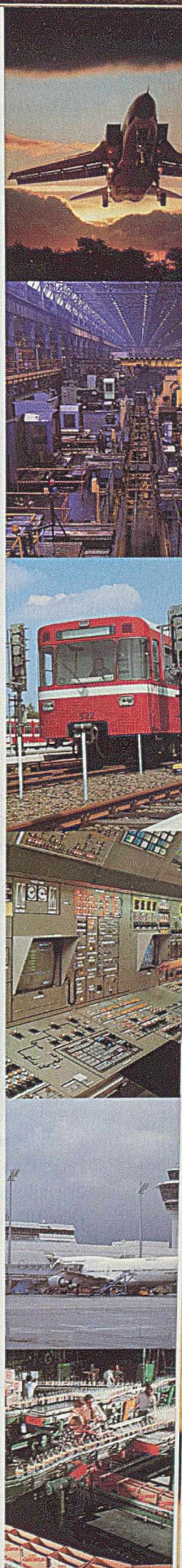
Maestro, take the baton! You are in the land that lies beyond the one-stop shop. Here's where competition for your business is based on a simple formula: technology, quality and service. You'll optimize your investment in systems and modular components that fullfill the requirements of today and tomorrow.

◆ Construct a concerto for a deterministic network-based system.

◆ Pen the notes for a very high performance physics lab computer that cannot grow old or weak.

◆ Compose a supervisory control and data acquisition system that also runs the factory controlled quality.

You'll love this place...



Organize the power

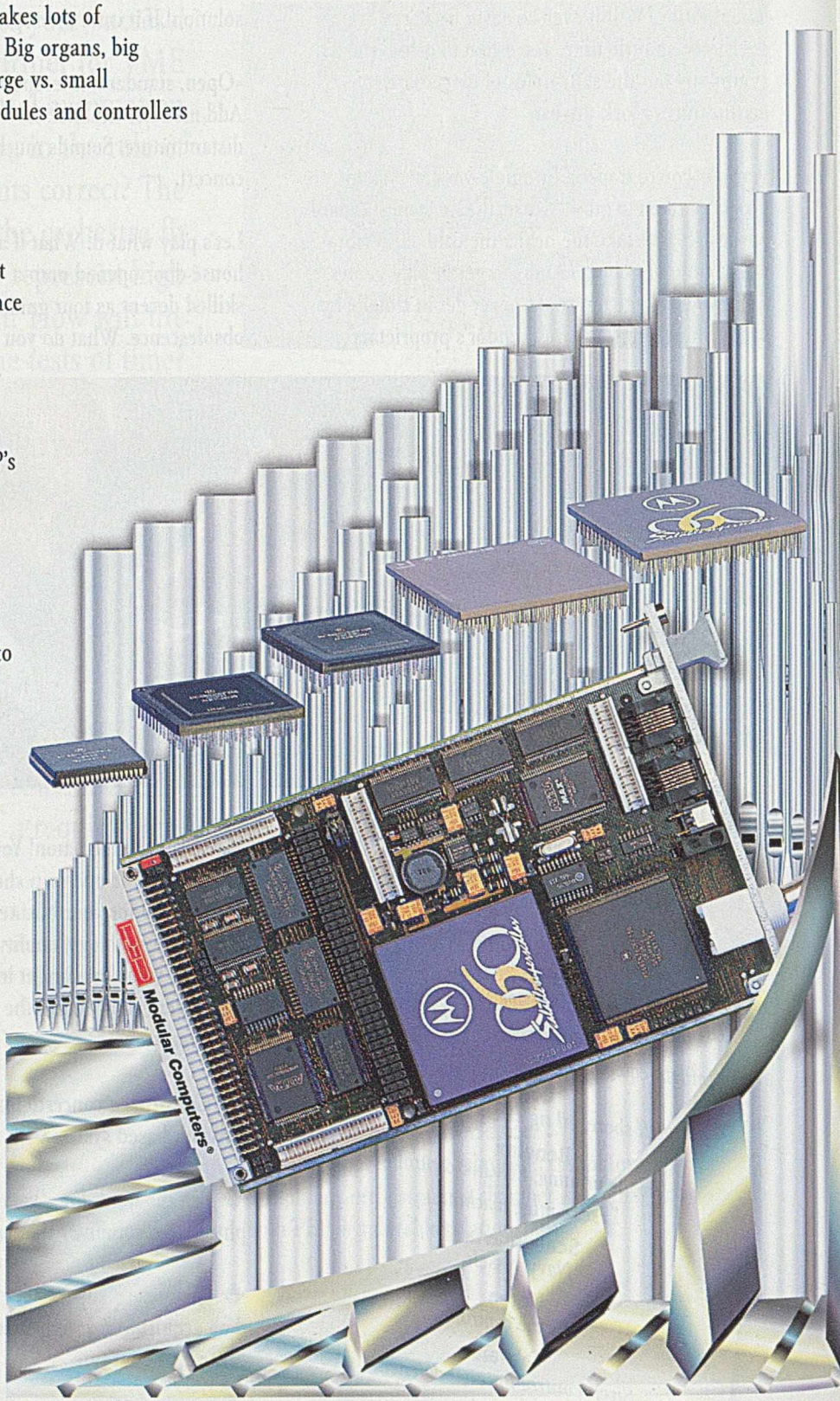
Applications are like orchestras. It takes lots of instruments to move large audiences. Big organs, big sound! Scale performance just like large vs. small computers. Add CPU boards, I/O modules and controllers to achieve the power you need.

Isn't this the core idea for open standards? Modularity: it's the key instrument of versatility and power. It is how you get exceptional performance for an expected price.

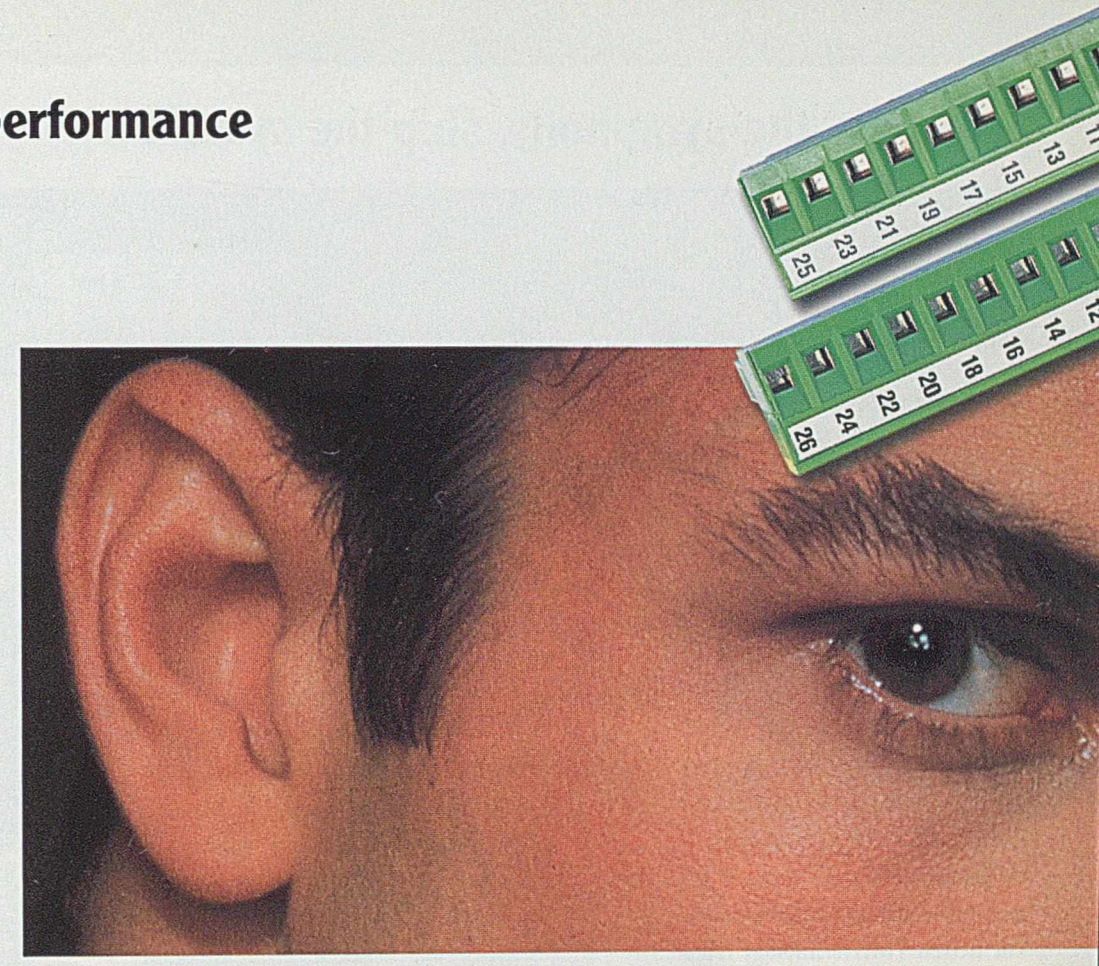
Industrial computing's most faithful open, standard, modular entity is the 68000 family of microprocessors. PEP's family. We take it to the top with the new VM62 68060-based single board computer.

It's just one of many single board engines. Power to enable customers to buy just what's needed, then, to expand. From 1 to 100 MIPs per board. More than 2000 MIPs per 21-board VMEbus system! You probably don't need this much... yet.

Someday, you may. That's why PEP invented the AutoBahn Spanceiver, a companion data mover that keeps the engines fed with data at 200 million bytes per second. One chip, 10X boost. That's power.



Energize the performance

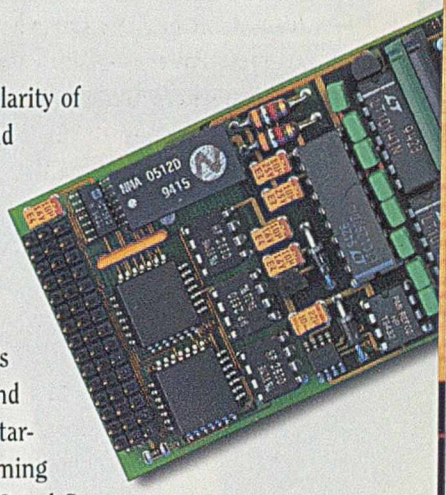


In the land beyond the one-stop shop, power isn't everything. Modularity is equally important.

Open system standards thrive on a hierarchy of modularity. Modules for I/O. For software. For connectivity to applications. The result is exceptional price/performance.

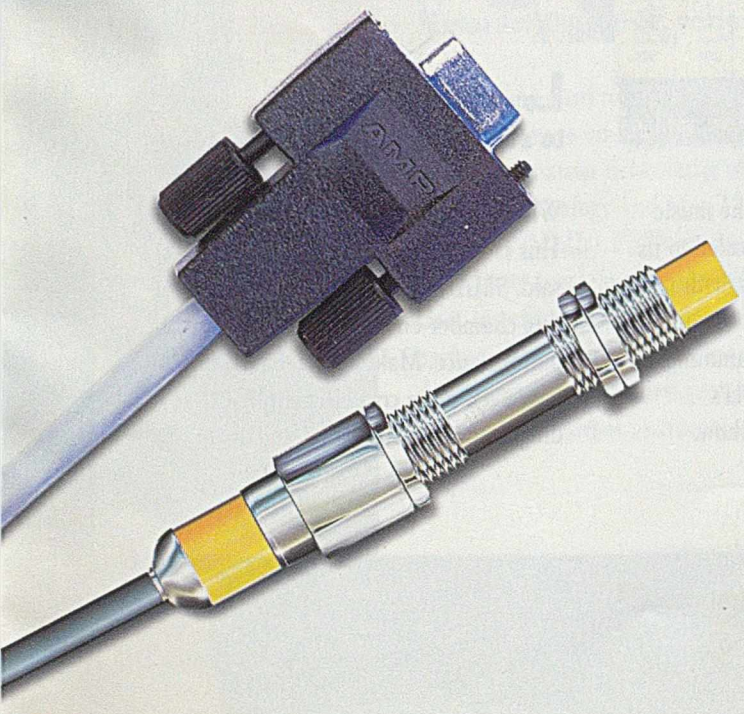
PEP's Controller eXtension Modules and Modpacks increase the functionality of the engines. PEP's industrial I/O modules with screw terminal interfaces ease the connection to the system.

The software must be modular so it can be layered to fit the need. Portable code crosses applications running on microkernels that control the engines. Microkernels that are de facto standards like OS-9, VxWorks , VRTX and Spectra , and pSOS .

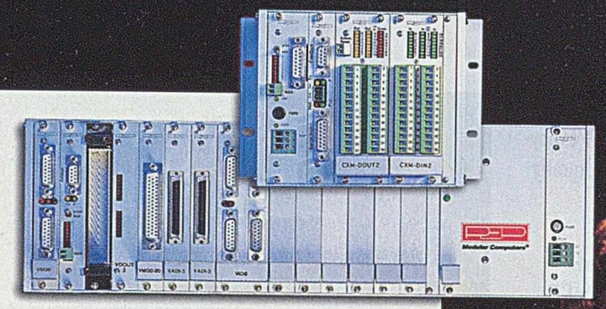


Thanks to the growing popularity of Windows-based programs and development tools for computer-aided software engineering, PEP`s modules incorporate UNIX- and Windows compatibility. The variety of tools is incredible: visualization and man-machine interfaces for target hardware; PLC programming for controls under IEC1131-3 and C; and development environments for UNIX and PC`s or self-hosted targets.

All the tools to energize any performance.



Complete the symphony, ship the system

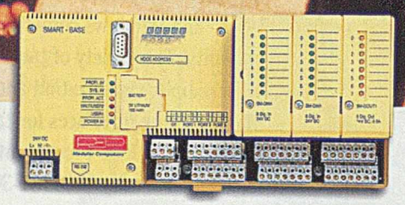


Comprehensive System Thinking

What expression describes a thoroughness so complete you want to see it, hear it, taste it, play it again? When the solution you most need is exactly what you get, and will be the next time even if the requirement changes, transparently-networked systems and software brings the solution.

Take PEP's VMEbus systems in 3- to 21-slot versions. With vast modularity, these are industry's tour-de-force solutions for real-time computing and automation. The richest VME technology remains contemporary thanks to Motorola's venerable 68xxx scalable architecture. PEP's systems host every form of industrial I/O (from PEP`s one-stop shop and other vendors). They run IEC1131-3 software and C-code in real time mode.

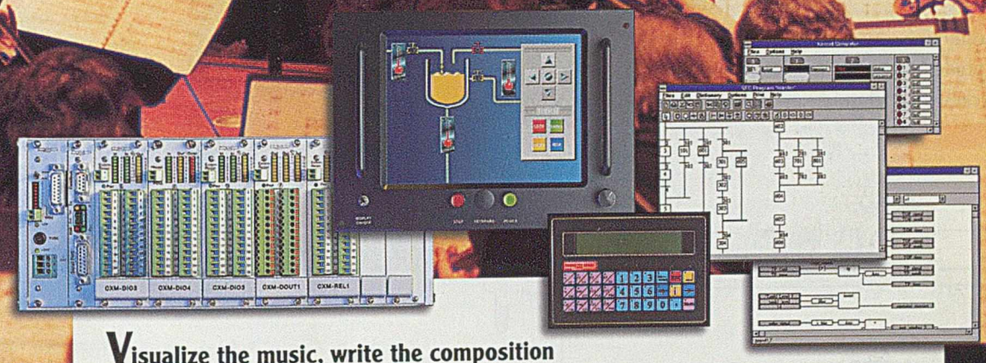
Any PEP system may transparently orchestrate activity on a real time fieldbus. Real- time standardized digital networking on the factory floor? You bet! Available now.



Play What-if again. What if the music of the PEP VME system was scaled to its most diminutive form? Busless, with no loss of appeal. Plus same I/O instruments. IEC1131-3 programming, a real-time fieldbus and Motorola's computer chip to conduct the show.

Low-cost gateway to standards

What's missing? Most of the cost! This is smart, you say. Exactly what we said. SMART I/O: the industrial computing chamber ensemble with concert hall performance. Makes VME based technology even more cost efficient by extending modularity down as well as up.



Visualize the music, write the composition

It used to be hard to see the factory floor unless you were on it. NO MORE, thanks to the transparent networking of digital fieldbus technology and Man-Machine interfaces (MMIs). PEP's hardware modularity combines a real time Graphical User Interface MGR and a visualisations package–to enable construction of fully deterministic designs that cannot be achieved on another architecture. Imagine: program any node from any node (including remote login from... home). That's transparent networking.

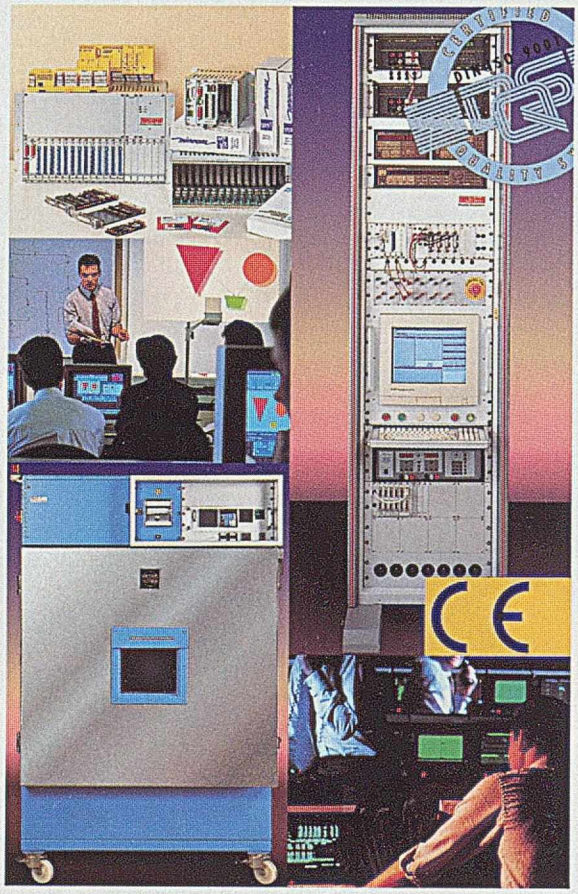
Prepare control code in native languages like Ladder Diagram, Sequential Function Charts or C. Simulate behavior. Debug and download on fieldbus for real-time operation. In no time, you'll complete the symphony and ship the target system.

The partnership you don't want to miss

System Packaging & Integration
The land beyond the one-stop shop offers the widest choices for backplanes, card cages and power supplies, as well as full custom packaging. Our application-specific modules provide anxiety-free customization with single part number convenience.

Educating the customer
Seminars and training sessions are produced with the same care as our boards and systems. We offer educational opportunities worldwide.

Environmental Qualifications
Extended temperature range capability and ruggedization are central to PEP's design cycle. PEP products survive harsh conditions because that's how we design them.



Testing All Products
PEP exercises all products with rigorous test cycles. This is especially important for analog I/O. We test extended temperature range operation as well as shock and vibration survival. PEP complies with CE, IEC and VDE requirements.

Supporting you on your way into the future
We offer 20 years of experience to you. Our support team will assist you in any way you need, training, software or system integration. PEP- a partner you can trust.

conception creative / V.Hilbel

X Response to PEP

YES I want to know more about PEP products. Please expedite your comprehensive catalog.

special interests are:

e

pany

t /Mail Stop

e / Fax

USA: PEP Modular Computers Inc. · 750 Holiday Drive, Building 9 · Pittsburgh, PA 15220
Tel.: ++ 1 412 921-3322 · Fax: ++ 1 412 921-3356 · Toll free 800-228-1737

UK: PEP Modular Computers Ltd · Unit 3 · Riverside Business Centre · Brighton Road
Shoreham-by-Sea · West Sussex · BN43 6RE · Tel.: ++ 44 (0) 1273 44 11 88 · Fax: ++ 44 (0) 1273 44 11 99

Sweden: PEP Modular Computers AB · Box 1430 · S-18314 Täby
Tel.: ++ 46 (0) 8 756 72 60 · Fax: ++ 46 (0) 8 732 63 10

Poland: PEP Poland · ul. Krcywickiego 9 · pokoj 1001/1002 Warscawa
Tel.: ++ 48 (22) 25 13 35 · Fax: ++ 48 (2) 25 13 35

PEP Modular Computers®

PEP Modular Computers GmbH · Apfeltrangerstr. 16 · D-87600 Kaufbeuren
Tel.: ++ 49 - (0) 83 41 - 803-0 · Fax: ++ 49 - (0) 83 41 - 803 499

automatowych. Z drugiej strony, struktura obiektowa zaciera strukturę przepływu sterowania między obiektami i nie ułatwia wyrażenia wymagań czasowych. Z tego względu, szczegółowy opis działania obiektu może obejmować, oprócz atrybutów i usług, również graf przejść automatu, opisującego zmiany stanu zachodzące podczas wykonywania usług. Ten sam obiekt może wykonywać różne usługi w różnych stanach.

Podanie ogólnego schematu postępowania charakteryzującego różne metody projektowania obiektowego jest trudne, gdyż metody te nie są jeszcze w pełni dojrzałe i silnie zależą od właściwości języka, w którym ma być realizowane oprogramowanie. Zasadnicza różnica występuje tu między metodami zorientowanymi na języki sekwencyjne, takie jak C++, i języki dopuszczające programowanie równoległe, takie jak Ada[2)].

W pierwszym wypadku (język sekwencyjny) schemat postępowania jest następujący:

1. Zbadaj dziedzinę zastosowania, znajdź występujące w tej dziedzinie obiekty i zbuduj model zależności między obiektami.
2. Określ atrybuty i usługi obiektów.
3. Podziel model między współbieżne zadania i uzupełnij szczegóły związane z komunikacją zadań z innymi zadaniami i ze światem zewnętrznym.
4. Dla każdego obiektu każdego zadania: określ strukturę atrybutów i algorytmy usług.
5. Dla każdego obiektu każdego zadania: określ strukturę programów realizujących usługi.

Pierwotną strukturą oprogramowania jest tu podział programów na zadania współbieżne. Podział sekwencyjnych programów zadań na obiekty jest strukturą wtórną [15].

W drugim wypadku (język równoległy) schemat postępowania jest inny:

1. Zbadaj dziedzinę zastosowania, znajdź występujące w tej dziedzinie obiekty i zbuduj model zależności między obiektami.
2. Określ atrybuty i usługi obiektów.
3. Zdefiniuj sposób współpracy obiektów.
4. Dla każdego obiektu: określ strukturę atrybutów i sposób realizacji usług: niektóre usługi mogą być dekomponowane między grupę zagnieżdżonych obiektów.
5. Dla każdego obiektu: określ strukturę programów realizujących usługi.

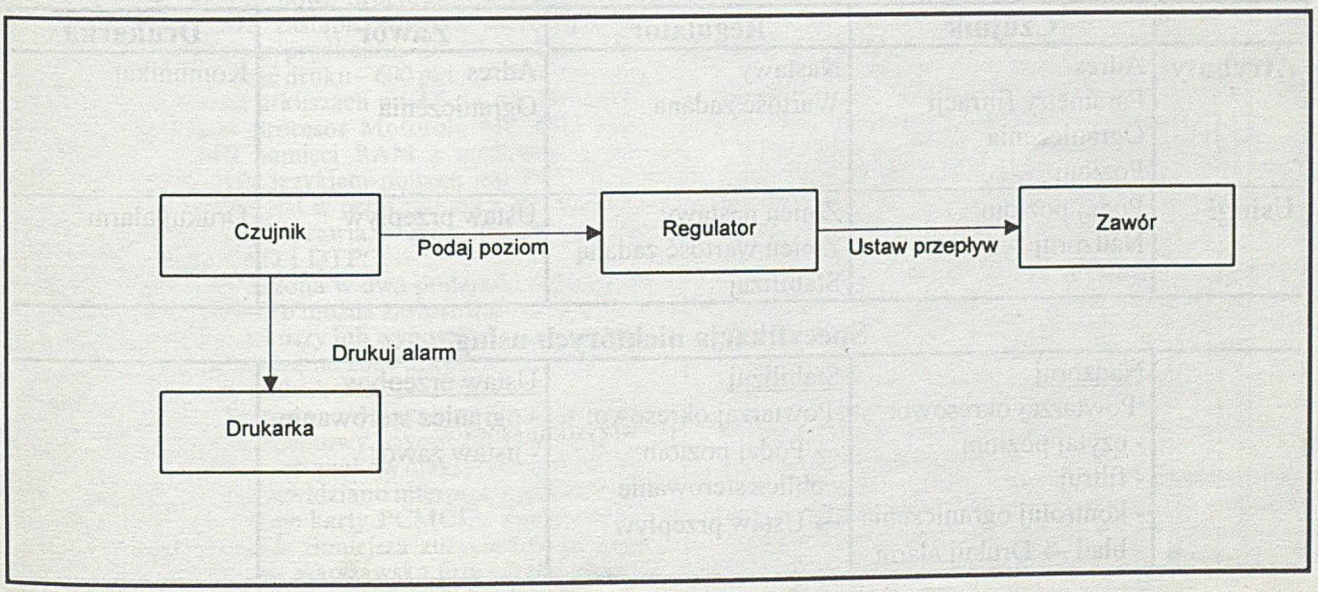
W tym wypadku, pierwotną strukturą oprogramowania pozostaje podział całości programów na obiekty. Wyodrębnienie zadań następuje wewnątrz obiektów, i jest strukturą wtórną [3].

Wróćmy do przykładu systemu sterującego poziomem cieczy w zbiorniku, rozważanego w poprzednim rozdziale. W dziedzinie zastosowania, tzn. w instalacji, można wyróżnić cztery obiekty: czujnik, regulator, zawór i drukarkę. Odwzorowanie obiektów rzeczywistych w obiekty programowe prowadzi do modelu obiektowego, przedstawionego na rys. 5. Warto zauważyć, że strzałki na tym rysunku nie pokazują kierunku przepływu danych, lecz kierunek wywoływania usług. Wymagane w instalacji przetwarzanie jest realizowane wewnątrz obiektów podczas wykonywania usług. Usługa *podaj poziom* realizuje pomiar, filtrację i kontrolę ograniczeń, a usługa *ustaw przepływ* realizuje ograniczenie sterowania i wyprowadzenie nastawy do zaworu. Opracowanie takiego modelu jest wynikiem analizy wymagań, wykonanej w pierwszym kroku metody.

Dalsze postępowanie zależy wyraźnie od rodzaju metody. W wariancie sekwencyjnym podział modelu między współbieżne zadania prowadzi do wyodrębnienia zadania regulacji (obiekty: *Czujnik, Regulator, Zawór*) i zadania drukowania (obiekty: *Drukarka*). Uwzględniając istnienie programu głównego, wywołującego usługi obiektów, i podprogramów, implementujących przetwarzanie realizowane przez usługi tych obiektów, otrzymuje się strukturę programu niemal identyczną ze strukturą przedstawioną na rys. 4. W odróżnieniu od sytuacji w poprzednim rozdziale, nie ma tu jednak gwałtownego przejścia od modelu obiektowego do modelu struktury programu. Obiekty pozostają w strukturze programu jako pewne kolekcje podprogramów, które wywołują się nawzajem,

[2)] Ada nie jest językiem obiektowym – nie ma w nim klas i dziedziczenia; są natomiast pakiety – jednostki programowe, które mogą zawierać struktury danych, podprogramy i zadania.

Rys. 5. Model obiektowy systemu sterującego poziomem cieczy w zbiorniku



realizując usługi zdefiniowane w modelu obiektowym. Struktura programu odzwierciedla naturalną strukturę dziedziny zastosowania.

Metody zorientowane na języki równoległe nie narzucają potrzeby sztucznego dzielenia modelu obiektowego na zadania. Równoległość przetwarzania może być ukryta wewnątrz obiektów i nie narusza struktury modelu. Definicje atrybutów i usług obiektów przykładowego systemu sterującego poziomem cieczy w zbiorniku są pokazane w tabeli.

Implementacyjna struktura programu, określona w czwartym kroku metody, składa się z czterech pakietów, reprezentujących cztery obiekty systemu. Pakiet *Czujnik* zawiera zmienne przechowujące wartości atrybutów, zadanie realizujące usługę *Nadzoruj* i ustalające cyklicznie wartość zmiennej *Poziom*, oraz podprogram *Podaj_poziom*, udostępniający aktualną wartość zmiennej *Poziom* innym pakietom. Pakiet *Regulator* zawiera zmienne przechowujące wartości atrybutów, zadanie realizujące usługę *Stabilizuj*, oraz podprogramy *Zmień_nastawy* i *Zmień_wartość_zadaną*, umożliwiające zmianę parametrów regulatora innym pakietom (na razie żaden pakiet nie korzysta z tych usług). Pakiety *Zawór* i *Drukarka* są strukturalnie prostsze i zawierają tylko atrybuty i podprogramy realizujące usługi.

Ostatni krok metody sprowadza się do określenia struktury wszystkich zadań i podprogramów. Na przykład, w programie zadania usługi *Nadzoruj* można wyróżnić trzy podprogramy: *Pomiar*, *Filtracja* i *Kontrola*.

Na zakończenie – tak jak w poprzednim rozdziale – rozważmy problem rozszerzenia systemu sterującego o dodatkowe funkcje graficzne. Tym razem zmiana będzie znacznie prostsza do wykonania niż poprzednio. Modyfikacja oprogramowania sprowadzi się do dodania nowego obiektu, np. *Wykres*, zawierającego wewnętrzne zadanie rysujące, i wywołującego usługę *Podaj_pomiar* obiektu *Czujnik*. Żadne zmiany w już istniejących programach nie są konieczne.

Ocena

Potencjalne zalety metod obiektowych są ogromne. Po pierwsze, metody te oferują jednolity model koncepcyjny i jednolitą notację, używane na przestrzeni całego projektu. Po drugie, oparcie struktury programów na strukturze dziedziny zastosowania gwarantuje dużą stabilność tej struktury i odporność na późniejsze zmiany i modyfikacje. Podstawowe charakterystyki obiektów odzwierciedlają podstawowe cechy obiektów fizycznych i nie są ściśle uzależnione od kontekstu konkretnego projektu. To z kolei sprzyja powtarzalnemu wykorzystywaniu raz opracowanych obiektów w różnych projektach. Po trzecie, wąski i dobrze określony sprzęg między współpracującymi obiektami (tylko wywołania usług) stwarza możliwość dokładnej kontroli tych wywołań, co sprzyja osiągnięciu dużej odporności modułów programu. Po czwarte, zasada dziedziczenia atrybutów i funkcji pozwala na bardzo szybkie definiowanie nowych obiektów, które stanowią szczególne przypadki obiektów już zdefiniowanych.

Do wad należy zaliczyć trudność specyfikowania i weryfikacji wymagań czasowych narzuconych na przetwarzanie zdarzeń zewnętrznych. Jeżeli przetwarzanie zdarzeń wymaga współdziałania wielu obiektów, to śledzenie ścieżki przetwarzania wymaga dokładnego wnikania w wewnętrzną strukturę obiektu i zmusza do opracowania pomocniczego modelu przepływu danych, przenikającego przez wszystkie współpracujące obiekty. Zalecenia budowy takiego modelu funkcjonalnego można znaleźć np. w [15].

Pełna ocena metod obiektowych w zastosowaniu do projektowania systemów sterujących nie jest jednak tak prosta, jak by to wynikało z podanego zestawienia zalet i wad. Problem polega na małej dojrzałości obecnie istniejących metod, które nie zostały jeszcze sprawdzone w dużych projektach przemysłowych, i na braku odpowiednich języków programowania.

Szczególną cechą wszystkich systemów sterujących jest bardzo silna równoległość działania elementów środowiska (instalacji sterowanej) oraz ostre wymagania dotyczące terminowości przetwarzania. Jak starałem się pokazać na przykładach, odwzorowanie równoległości środowiska w obiektowej strukturze programu wymaga zastosowania obiektowego języka programowania, zorientowanego

Obiekty i usługi

	Czujnik	Regulator	Zawór	Drukarka
Atrybuty	Adres Parametry filtracji Ograniczenia Poziom	Nastawy Wartość zadana	Adres Ograniczenia	Komunikat
Usługi	Podaj poziom Nadzoruj	Zmień nastawy Zmień wartość zadaną Stabilizuj	Ustaw przepływ	Drukuj alarm
Specyfikacja niektórych usług				
	Nadzoruj Powtarzaj okresowo: - czytaj poziom - filtruj - kontroluj ograniczenia - błąd → Drukuj alarm	Stabilizuj Powtarzaj okresowo: → Podaj poziom - oblicz sterowanie → Ustaw przepływ	Ustaw przepływ - ogranicz sterowanie - ustaw zawór	

na przetwarzanie równoległe. Jedyny naprawdę popularny język obiektowy – C++ – nie spełnia tego wymagania. Stąd metody obiektowe są w najszerszym zakresie stosowane w kręgu projektantów i programistów języka Ada, który jest najbardziej zgodny z duchem tej metodologii.

Warto również zauważyć, że w projektowaniu systemów sterujących nie ma wielu okazji do wykorzystania zalet definiowania obiektów przez dziedziczenie. Obiekty technologiczne są w większości niepowtarzalne i nie są szczególnymi przypadkami jakichś innych, ogólniejszych obiektów. Jest to sytuacja różna od sytuacji, która występuje np. przy programowaniu okienkowego sprzęgu programu z użytkownikiem albo przetwarzaniu listowych, grafowych i drzewiastych struktur danych. W tych dziedzinach zastosowania występuje wiele standardowych obiektów, tworzących długą hierarchię uszczegółowienia, w których możliwość szybkiego powoływania nowych obiektów przez dziedziczenie i dopasowywania ich usług przez przeciążanie jest wprost nieoceniona. W tych też dziedzinach najszybciej rośnie popularność sekwencyjnego języka obiektowego C++.

LITERATURA

[1] Balzer R., Cheatham T.E., Green C.: Software Technology in the 1990s: Using a New Paradigm, Computer, Vol. 16, No. 11, pp. 39–45, 1983

[2] Boehm B. i in.: Characteristics of Software Quality, TRW Series of Software Technology, North-Holland Publishing Co., Amsterdam, 1978

[3] Booch G.: Object-oriented Development, IEEE Trans. Software Eng., Vol. SE-12, No. 2, pp. 211–221, 1986

[4] McCall J. (ed.): Factors in software quality, General Electric, 1977

[5] Gomaa H.: A Software Design Method for Real-Time Systems, Commun. ACM, Vol. 27, No. 9, pp. 938–949, 1984

[6] Harel D. i in.: STATEMATE: A Working Environment for the Development of Complex Reactive Systems, IEEE Trans. Software Eng., Vol. 16, No. 4, pp. 403–413, 1990

[7] Hatley D.J., Pirbhai I.A.: Strategies for Real-Time System Specification, Dorset House, 1987

[8] HOOD Working Group, HOOD Reference Manual. Issue 3.0, European Space Agency, Noordwijk, The Netherlands, 1989

[9] Huber P., Jensen K., Shapiro R.: Hierarchies in Coloured Petri Nets, w: Rozenberg G. (ed): Advances in Petri Nets 1990, LNCS 483, pp. 313–342, Springer, 1991

[10] Jackson M.A.: System Development, Prentice Hall, Englewood-Cliffs, 1983

[11] Jones C.B.: Software Development: A Rigorous Approach, Prentice-Hall International, 1980

[12] Meyer B.: Object-oriented Software Construction, Prentice-Hall, Englewood Cliffs, 1988

[13] Ramamoorthy C.V., Prakash A., Tsai W-T., Usuda Y.: Software Engineering: Problems and Perspectives, w: Software Engineering Project Management, IEEE, pp. 57–74, 1990

[14] Ross D.T.: Applications and Extensions of SADT, Computer, Vol. 18, No. 1, pp. 25–34, 1985

[15] Rumbaugh J., Blaha M., Premerlani W., Eddy S., Lorensen W.: Object-oriented Modelling and Design, Prentice Hall, Englewood-Cliffs, 1991

[16] Sievert G.E., Mizell T.A.: Specification-based Software Engineering with TAGS, Computer, Vol. 18, No. 4, pp. 56–65, 1985

[17] Simpson H.: The Mascot Method, Software Eng. Journal, Vol. 1, No. 3, pp. 103–120, 1986

[18] Ward P.T., Mellor S.J.: Structured Development for Real-Time Systems, Prentice-Hall, Englewood-Cliffs, 1985

[19] Williams J.: Systems Building Made Simple, Systems International, Nov. 1987

[20] Zave P.: An Operational Approach to Requirements Specification for Embedded Systems, IEEE Trans. Software Eng. Vol. SE-8, pp. 250–269, May 1982.

Fujitsu proponuje

Print Partner 30

• Może zastąpić kserokopiarkę

Print Partner 30 – to nowa drukarka *heavy duty* firmy Fujitsu, która może zastąpić również kserokopiarkę. Urządzenie drukuje z prędkością do 30 str./min. Maksymalna rozdzielczość druku – 600 pkt./cal dwustronnie, można pracować na arkuszach od A5 do A3. Drukarka jest wyposażona w procesor Motorola MC 68EC040–24MHz i 16 MB pamięci RAM z możliwością rozszerzenia do 64 MB. Językiem poleceń jest PCL 5e, co sprawia, że drukarka jest w pełni zgodna z HP LaserJet IV; emulacja HP GL/2 sprawia, że może być używana do prac graficznych, CAD i DTP.

Drukarka jest wyposażona w dwa podajniki papieru po 500 arkuszy, dodatkowo można zastosować trzeci podajnik również na 500 arkuszy lub wyposażyć w zewnętrzny podajnik i tacę odbiorczą na 2000 arkuszy. Jednorazowo można bez przerwy zadrukować do 3500 arkuszy. Print Partner 30 ma wbudowane 45 skalowalne kroje pism i specjalny język dokumentowy Layout 2.0. Dostępny jest pełny zakres europejskich znaków diakrytycznych, w tym polskie. Jako opcję przewidziano interpreter języka PostScript (Level 2) w formie karty PCMCIA. Zastosowanie trybu pracy *economode* zmniejsza zużycie tonera. Dystrybutorem Fujitsu jest warszawska firma Initel. **(k)**

Power PC na biurku i w teczce...

Nowości IBM

• Rozmaite konfiguracje

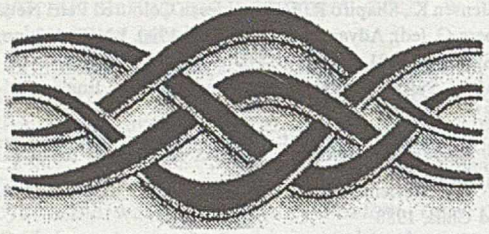
IMB wprowadził do sprzedaży nową serię komputerów osobistych i przenośnych opartych na procesorach PowerPC – IBM Personal Computer Power Series i ThinkPad Power Series. Mogą one pełnić rolę *super klienta,* reprezentowaną do tej pory przez najpotężniejsze stacje robocze. Wyposażone są w narzędzia komunikacyjne najwyższej jakości oraz zaawansowane interfejsy użytkownika, upraszczające współpracę sieciową.

Komputery są dostępne w rozmaitych konfiguracjach, poczynając od IBM PC Power Series 830, z RISC-owym mikroprocesorem PowerPC 604 z zegarem o częstotliwości 100 MHz, aż po IBM PC Power Series 850 z PowerPC 604 z zegarem 133 MHz. W komputerach tych można będzie w przyszłości wymieniać mikroprocesory na jeszcze szybsze. Do notebooków ThinkPad Power Series 850 wprowadzono standard G10 Graphics ze sterownikiem Motion Video Adapter dla wejścia i wyjścia głosu, z wykorzystaniem standardowych modemów i koprocesora w procesorze PowerPC 603e. Standardowo notebooki z nowych serii są wyposażane w oprogramowanie do przetwarzania sygnałów.

Wszystkie komputery klasy PC i notebooki z nowych serii są wyposażane w Sensory Suite, pakiet oprogamowania wykorzystujący możliwości wykonywania działań zmiennoprzecinkowych na RISC-owych PowerPC. **(k)**

Microsoft

Windows NT i systemy czasu rzeczywistego (cz. 1)



Istnieje wiele definicji systemu czasu rzeczywistego. Zanim zaczniemy omawiać zastosowanie Windows NT w tego typu systemach, zapoznajmy się z najbardziej popularną ich definicją: "System czasu rzeczywistego to taki system, w którym poprawność obliczeń zależy nie tylko od ich logicznej zgodności, ale również od czasu w jakim zostają wygenrowane. Jeśli limit czasowy zostaje przez system przekroczony, to uważa się, że wystąpił błąd systemowy."

Oznacza to, że systemy czasu rzeczywistego można podzielić na dwie podstawowe kategorie: - te które muszą spełniać rygorystyczne wymogi czasowe i te, dla których limit czasowy jest nieco bardziej elastyczny. Systemy pierwszej kategorii muszą bezbłędnie dostarczać odpowiedzi na pewnego rodzaju zdarzenia w określonym przedziale czasowym. Odpowiedź ta musi być przewidywalna i niezależna od innych czynności podejmowanych przez system operacyjny na bzaie innych zadań. Dostarczanie takiej odpowiedzi oznacza, że wywołania systemowe muszą mieć określone i mierzalne czasy reakcji. Systemy czasu rzeczywistego z tej kategorii często **wymagają specjalnych rozwiązań sprzętowych** ze specjalnymi sterownikami.

Używając tej definicji Microsoft Windows NT nie jest systemem czasu rzeczywistego pierwszej kategorii. Jest to system operacyjny ogólnego zastosowania wyposażony w możliwości bardzo szybkich czasów reakcji na zdarzenia - nie są one jednak aż tak deterministyczne jak w systemach czasu rzeczywistego pierwszej kategorii.

Jednakże rozpatrując drugą kategorię - zauważamy,że dotyczy ona systemów operacyjnych, które mają mniejsze wymogi dotyczące "niezawodnych opóźnień", ale wciąż muszą pracować bardzo szybko z zadowalającymi ramami czasowami reakcji na zdarzenia. Oznacza to, że system operacyjny musi być "wystarczająco dobry" na obsługę zdarzeń w zadowalającym czasie. Windows NT spełnia te kryteria.

Podsumowując - istnieje **bardzo niewiele aplikacji**, które koniecznie wymagają systemów operacyjnych czasu rzeczywistego pierwszej kategorii. Większość z nich potrzebuje systemów "wystarczająco dobrych" do spełnienia określonych kryteriów. W takich sytuacjach system operacyjny Windows NT jest najlepszym systemem operacyjnym dla tego typu zadań.

Firma Microsoft w porozumieniu z firmami obsługującymi rynki systemów czasu rzeczywistego wystąpiła z dwoma inicjatywami. Dla klientów z rynku usług finansowych, na którym analitycy finansowi i handlowcy muszą niemal natychmiast znać wszystkie ruchy cen i towarów przeznaczona jest inicjatywa o nazwie OLE for Real Time Market Data. Ma ona za zadanie dostarczać standardowego sposobu dostępu do danych finansowych dla ludzi z tej branży.

Natomiast dla naukowców i producentów, Microsoft we współpracy z ludźmi z tej branży opracował zestaw interfejsów do Windows pozwalających na bezproblemową integrację systemów zbierania danych. Inicjatywa ta nosi nazwę Windows for Science, Engineering and Manufacturing, w skrócie WINSEM.

W drugiej części artykułu opiszemy dokładnie podstawowe elementy systemu Windows NT mające bezpośredni związek z pracą w czasie rzeczywistym. Będą to m.in: reagowanie na zdarzenia zewnętrzne, priorytety i rozdzielanie zadań systemowych, synchronizacja wątków i procesów oraz deterministyczny czas odpowiedzi.

0/1/95

Obiektowe podejście do modelowania i analizy komputerowych systemów sterowania

Janusz Górski, Bartosz Nowicki
Francusko-Polska Wyższa Szkoła Nowych Technik Informatyczno-Komunikacyjnych
Poznań

W artykule przedstawiono podejście obiektowe do analizy komputerowych systemów sterowania. W pierwszym kroku tworzony jest model obiektowy rozpatrywanego systemu. Następnie budowane są modele dynamiczne obiektów składowych. Modele te są przedmiotem analiz, weryfikujących funkcje systemu, przewidziane jego misją. Następnie, na podstawie zidentyfikowanych stanów hazardowych jest tworzony model systemu rozróżniający stany bezpieczne i niebezpieczne. Model taki stanowi punkt wyjścia dla analizy bezpieczeństwa oraz ewentualnych modyfikacji, mających na celu zapobieganie sytuacjom niebezpiecznym. Prezentacji omawianego podejścia dokonano w kontekście przykładowego systemu aplikacyjnego – sterowania spalaniem w palniku gazowym.

Wiele współczesnych dziedzin działalności przemysłowej i usługowej wiąże się ze znacznym ryzykiem, np. przemysł chemiczny, energetyka, transport, medycyna, itp. Zastosowanie w tych dziedzinach informatyki powoduje, że znaczna (a często zasadnicza) część odpowiedzialności związanej z utrzymaniem tego ryzyka w akceptowalnych granicach przenosi się na systemy komputerowe, a w szczególności na zawarte w nich oprogramowanie. Powoduje to zdecydowany wzrost znaczenia *wiarygodności* systemów komputerowych. W ujęciu ogólnym wiarygodność określa stopień zaufania do usług świadczonych przez system komputerowy umieszczony w kontekście aplikacyjnym. W interpretacji węższej – i bardziej technicznej – drogą do zapewnienia wiarygodności jest uzyskanie gwarancji, że system komputerowy ma określone własności, których zachowanie wyklucza lub ogranicza niektóre elementy ryzyka, związanego z danym zastosowaniem.

W prezentowanym artykule przedstawiono podejście, polegające na modelowaniu systemu komputerowego w jego kontekście aplikacyjnym. Model ten jest podstawą do sformułowania (pożądanych) własności prowadzących do eliminacji ryzyka, związanego z zastosowaniem systemu. Dzięki zastosowaniu modelowania formalnego uzyskano możliwość precyzyjnej analizy, mającej na celu uzyskanie odpowiedzi na pytanie, czy własności te są zachowane przez rozpatrywany system.

Powyższe podejście wymagało rozwiązania następujących problemów:

- Wybór modelu systemu i upewnienie się, że adekwatnie reprezentuje on rozpatrywaną rzeczywistość. Zastosowano tu modelowanie obiektowe, którego zaletą jest znaczna zgodność struktury modelu z rzeczywistością podlegającą modelowaniu.
- Specyfikacja i analiza pożądanych własności systemu. Przez wykorzystanie podejścia obiektowego uzyskano możliwość modelowania systemu wraz z jego szerokim kontekstem aplikacyjnym. Umożliwiło to wyrażenie (w terminach modelu) wymagań zewnętrznych odnoszących się do środowiska, w jakim istnieje rozpatrywany system komputerowy. Dzięki temu można było sformułować warunki bezpieczeństwa systemu oraz dokonać analizy ich spełnialności.

Prezentacji dokonano na przykładzie zadania sterowania palnikiem gazowym. Model powyższego problemu poddano analizie, w ramach której przeprowadzono dwa rodzaje weryfikacji:

- Weryfikację wymagań funkcjonalnych, dającą odpowiedź na pytanie czy system jest zdolny do wypełnienia swojej misji, np. pytania typu:

czy (jeżeli nie wystąpi przeciąg) gaz zostanie zapalony na polecenie operatora?
czy nie nastąpi samoczynny zapłon gazu, bez polecenia operatora? itd.

Tego typu weryfikacja wykazuje, że mamy do czynienia z „właściwym" modelem, który odpowiada celom funkcjonalnym stawianym przed systemem.

- Weryfikację wymagań bezpieczeństwa, która daje odpowiedź na pytanie czy system nie doprowadzi do stanu niebezpiecznego – w rozpatrywanym kontekście chodzi o nadmierne nagromadzenie się gazu i zagrożenie wybuchem.

Do opisu własności dynamicznych systemu zastosowano notację *statechart-ów* [2–5]. Dzięki sformalizowanej semantyce tej notacji uzyskano możliwość precyzyjnej analizy własności dynamicznych modelu. Analizę przeprowadzono z wykorzystaniem narzędzia STATEMATE [6].

W artykule przedstawiono model systemu palnika gazowego utworzony przy użyciu metody obiektowej *Object Modelling Technique* [1]. Dokonano identyfikacji zagrożeń w rozpatrywanej aplikacji (wybuch gazu). Następnie dokonano analizy modelu pod kątem możliwości powstania takiego zagrożenia.

Model obiektowy systemu

Analizowany system steruje palnikiem gazowym, który dostarcza ciepła do pewnego procesu technologicznego. Podstawowym elementem systemu jest palnik z doprowadzonym do niego dopływem gazu oraz układem zapłonowym. Czujnik temperatury pozwala na potwierdzenie procesu spalania gazu. Zawór gazowy oraz zapłon są sterowane przez moduł sterujący (komputer). Do tego modułu jest doprowadzony zewnętrzny sygnał włączania i wyłączania spalania.

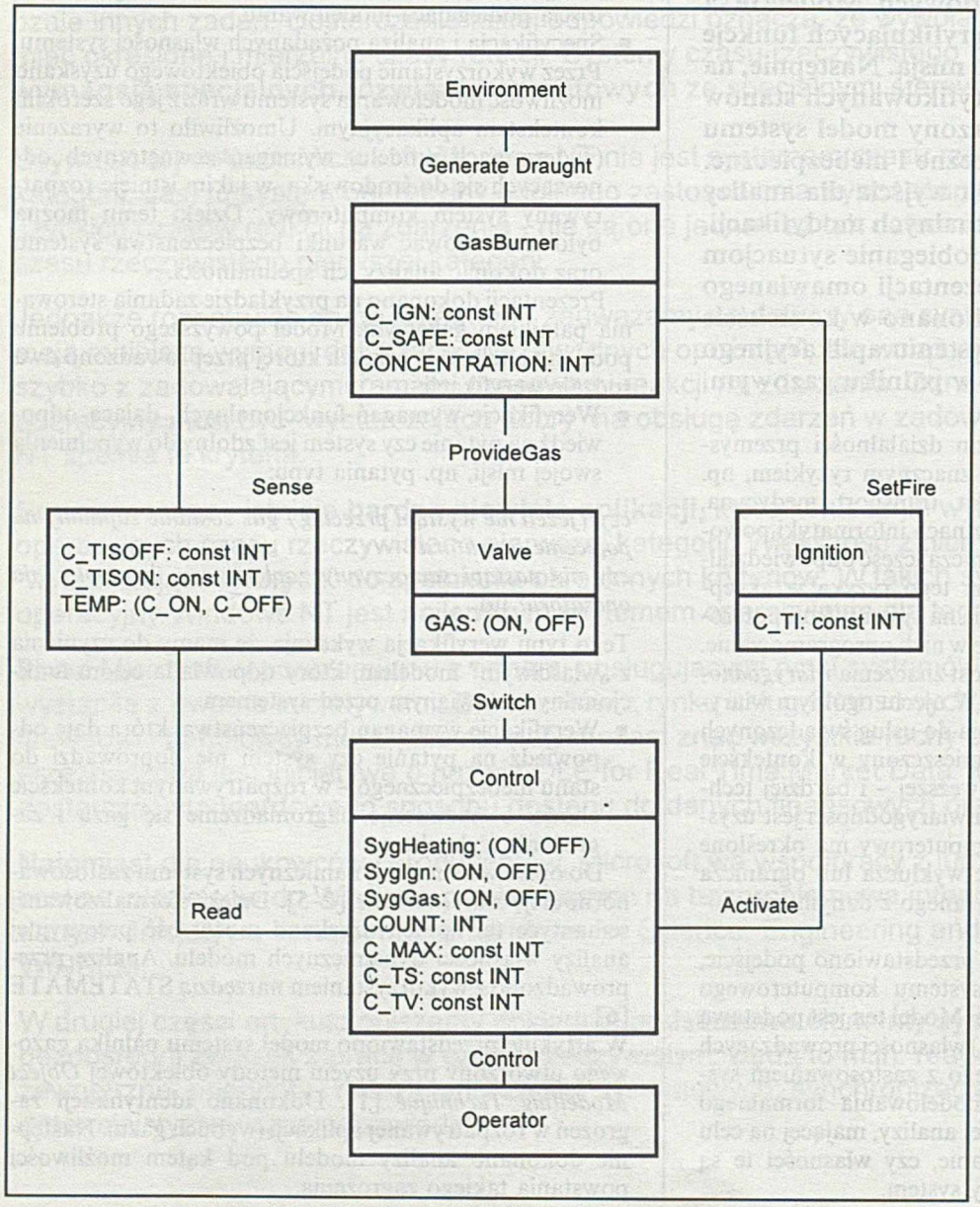
Zakładamy, że stężenie gazu w komorze spalania palnika jest wprost proporcjonalne do czasu otwarcia zaworu gazu (w przypadku braku iskry zapłonowej) i odwrotnie proporcjonalne do czasu przy zamkniętym zaworze (wietrzenie). Zadane jest również minimalne stężenie gazu, przy którym iskra może doprowadzić do zapalenia płomienia. Zapalniczkę, generującą iskrę, charakteryzuje zadany okres ładowania. Zapalniczka ignoruje polecenia generacji iskry, które otrzymuje podczas ładowania. Czujnik temperatury cechuje się pewną bezwładnością.

Nie każda iskra pojawiająca się w obecności odpowiedniej ilości gazu doprowadza do zapłonu. W pewnych przypadkach może dojść do zgaszenia płomienia (np. na skutek przeciągu). Zakłada się, że podczas eksploatacji systemu gaz jest zawsze dostępny.

Do opisu powyższego systemu zastosowano metodę modelowania obiektowego OMT [1]. Podstawą opisu systemu są trzy modele:

- *model obiektowy*, który pokazuje podstawowe zależności strukturalne w rozpatrywanym problemie aplikacyjnym, głównie związane ze związkami *całość-część, specjalizacja-uogólnienie, klasa-reprezentant klasy* oraz inne zależności o trwałym charakterze;
- *model dynamiczny*, który umożliwia opis zachowań poszczególnych obiektów. Zachowania te są wyrażane w terminach skończonej liczby stanów oraz przejść pomiędzy tymi stanami (wykorzystywana jest tu notacja *statechart-ów*);
- *model funkcjonalny*, który uwidacznia sposób przetwarzania danych w systemie; model ten pokazuje przepływy danych oraz funkcje odpowiedzialne za przekształcanie tych danych.

Rys. 1. Model obiektowy palnika gazowego



Model obiektowy rozpatrywanego systemu palnika został przedstawiony na rys. 1, natomiast w tabeli zamieszczono opis obiektów i ich atrybutów.

Model zawiera zarówno obiekty, składające się na *system sterujący* (Control), jak i obiekty, tworzące *system sterowany* (GasBurner, TempSensor, Ignition, Valve). Dodatkowo wprowadzono do modelu również te elementy *środowiska zewnętrznego* (Environment, Operator), które mogą wywierać wpływ na zachowanie się systemu. Utworzony w ten sposób model jest *zamknięty* – w tym sensie, że obejmuje wszystkie elementy brane pod uwagę podczas analizy. Odpowiedź na pytanie: czy zakres modelu jest dostateczny? – jest zadaniem walidacyjnym i wymaga rzetelnej analizy pod kątem zgodności modelu z rozpatrywanym zagadnieniem aplikacyjnym. W tym artykule nie będziemy się dalej zajmować tym problemem. Zauważmy jedynie, że uwidocznione związki pomiędzy obiektami reprezentują zależności występujące w rzeczywistym systemie, mające znaczenie podczas realizacji misji systemu:

- Czujnik temperatury bada (*Sense*) spalanie gazu.
- Wartości wskazywane przez czujnik są odczytywane (*Read*) przez układ sterujący.

- Układ sterujący przełącza (*Switch*) zawór gazu, który z kolei dostarcza gaz (*ProvideGas*) do palnika.
- Układ sterujący włącza (*Activate*) zapalniczkę, która zapala gaz (*SetFire*) znajdujący się w komorze spalania palnika.
- Operator wydaje komendy systemowi sterującemu (*Control*).
- Środowisko ingeruje w pracę instalacji gasząc płomień (*Genarate Draught*).

Model dynamiczny systemu

Model dynamiczny opisuje zachowanie się każdego obiektu, występującego w modelu obiektowym. Na rys. 2 przedstawiono *diagram zdarzeń* i *danych*, który prezentuje przepływy pomiędzy obiektami w systemie. Zdarzenia są reprezentowane liniami przerywanymi. Linie ciągłe reprezentują przepływ danych (odczyt atrybutu obiektu dokonywany przez inny obiekt). Diagram przepływu zdarzeń i danych dla palnika gazowego. Modele zachowań dla obiektów systemu palnika gazowego przedstawiono na rysunkach 3, 4, 5, 6 i 17.

Początkowo zapalniczka ładuje się (CHARGING). Po upływie czasu *C_TI* następuje przejście do stanu *gotowy* (READY). W tym stanie zapalniczka przebywa aż do wystąpienia zdarzenia SYGIGNON. Następuje wtedy przejście do stanu CHARGING z równoczesnym wygenerowaniem iskry (SPARK).

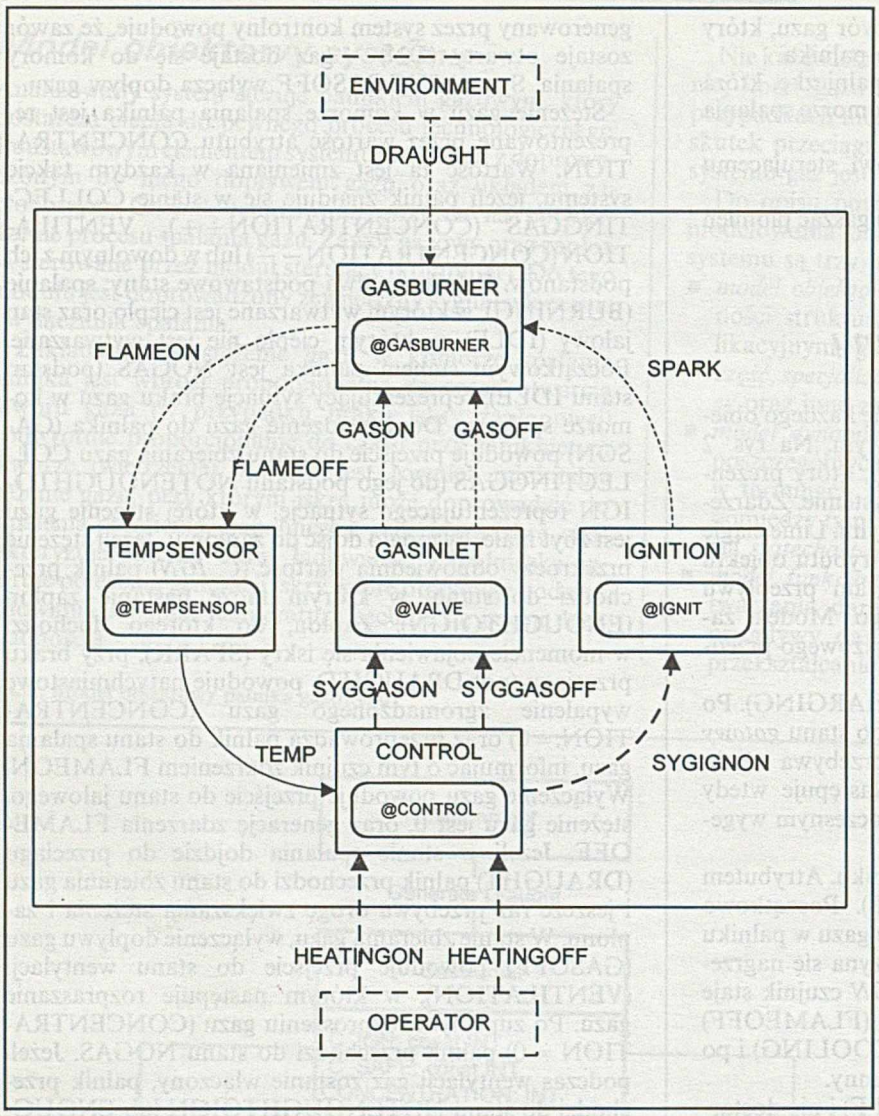
Czujnik wykrywa stan spalania w palniku. Atrybutem czujnika jest jego temperatura (TEMP). Początkowo czujnik jest zimny (COLD). Zapalenie się gazu w palniku (FLAMEON) powoduje, że czujnik zaczyna się nagrzewać (WARMING). Po okresie *C_TTSON* czujnik staje się gorący (HOT). Zgaszenie płomienia (FLAMEOFF) powoduje, że czujnik zaczyna stygnąć (COOLING) i po czasie *C_TTSOFF* staje się ponownie zimny.

Początkowo zawór jest zamknięty (OFF) i nie dostarcza gazu do komory spalania. Sygnał SYGGASON generowany przez system kontrolny powoduje, że zawór zostaje otwarty (ON) – gaz dostaje się do komory spalania. Sygnał SYGGASOFF wyłącza dopływ gazu.

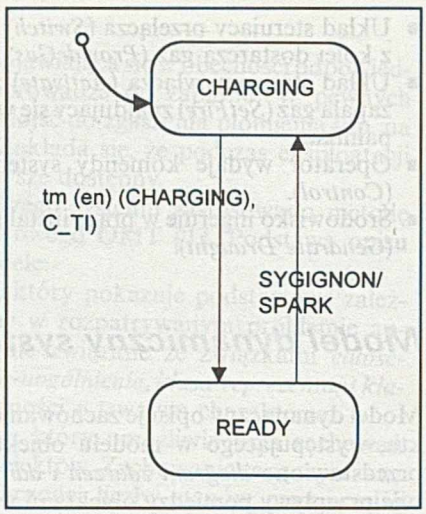
Stężenie gazu w komorze spalania palnika jest reprezentowane przez wartość atrybutu CONCENTRATION. Wartość ta jest zmieniana w każdym takcie systemu, jeżeli palnik znajduje się w stanie COLLECTINGGAS (CONCENTRATION++), VENTILATION (CONCENTRATION−−) lub w dowolnym z ich podstanów. Palnik ma dwa podstawowe stany: spalanie (BURNING), w którym wytwarzane jest ciepło oraz stan jałowy (IDLE), w którym ciepło nie jest wytwarzane. Początkowym stanem palnika jest NOGAS (podstan stanu IDLE), reprezentujący sytuację braku gazu w komorze spalania. Doprowadzenie gazu do palnika (GASON) powoduje przejście do stanu zbierania gazu COLLECTINGGAS (do jego podstanu NOTENOUGHTOIGN reprezentującego sytuację, w której stężenie gazu jest zbyt małe, by mogło dojść do zapłonu). Jeżeli stężenie przekroczy odpowiednią wartość (*C_IGN*) palnik przechodzi do stanu, w którym może nastąpić zapłon (ENOUGHTOIGN). Zapłon, do którego dochodzi w momencie pojawienia się iskry (SPARK), przy braku przeciągu (not DRAUGHT), powoduje natychmiastowe wypalenie zgromadzonego gazu (CONCENTRATION:=0) oraz przeprowadza palnik do stanu spalania gazu, informując o tym czujnik zdarzeniem FLAMEON. Wyłączenie gazu powoduje przejście do stanu jałowego, stężenie gazu jest 0, oraz generację zdarzenia FLAMEOFF. Jeżeli w stanie spalania dojdzie do przeciągu (DRAUGHT) palnik przechodzi do stanu zbierania gazu i jeszcze raz przebywa drogę zwiększania stężenia i zapłonu. W stanie zbierania gazu, wyłączenie dopływu gazu (GASOFF) powoduje przejście do stanu wentylacji (VENTILATION), w którym następuje rozpraszanie gazu. Po zupełnym rozproszeniu gazu (CONCENTRATION = 0) palnik przechodzi do stanu NOGAS. Jeżeli podczas wentylacji gaz zostanie włączony, palnik przechodzi do stanu NOTENOUGHTOIGN lub ENOUGHTOIGN, w zależności od aktualnego stężenia.

Opis obiektów i ich atrybutów

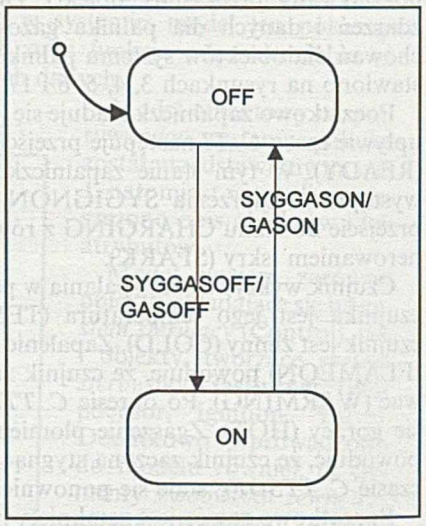
OBIEKT	ATRYBUT	OPIS
GasBurner		palnik gazowy
	C_IGN	minimalna ilość gazu w obrębie palnika, przy której może nastąpić zapłon
	C_SAFE	maksymalna bezpieczna ilość gazu
	CONCENTRATION	stężenie gazu
TempSensor		czujnik temperatury
	C_TTSOFF	czas chłodzenia czujnika
	C_TTSON	czas nagrzewania czujnika
	TEMP	stan czujnika
Valve		zawór gazu
	GAS	aktualny stan zaworu
Ignition		zapalniczka
	C_TI	czas ładowania zapalniczki
Control		układ sterujący spalaniem gazu (komputer sterujący)
	SygHeating	zewnętrzny sygnał włączenia/wyłączenia palnika
	SygIgn	sygnał sterowania zapalniczką
	SygGas	sygnał sterowania zaworem gazu
	COUNT	licznik pętli zapłonu
	C_MAX	maksymalna wartość licznika pętli zapłonu
	C_TS	czas synchronizacji (zapobiega wyścigom czasowym)
	C_TV	czas wentylacji palnika
Enviroment		obiekt odpowiedzialny za zewnętrzne gaszenie płomienia (generacja zdarzenia DRAUGHT).
Operator		obiekt sterujący instalacją



Rys. 2. Diagram zdarzeń i danych

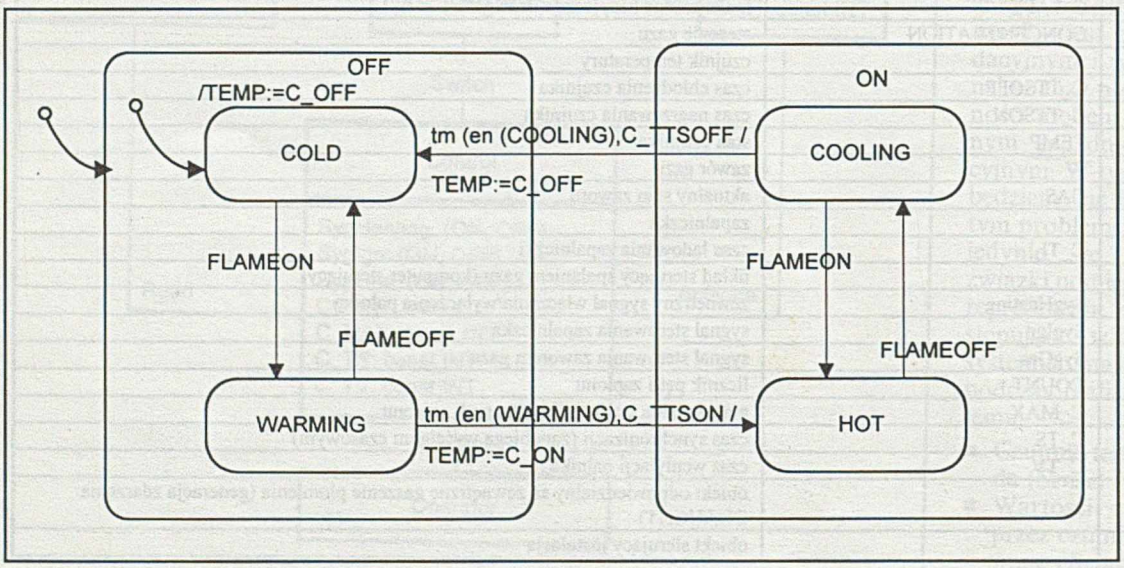


Rys. 3. Model dynamiczny zapalniczki

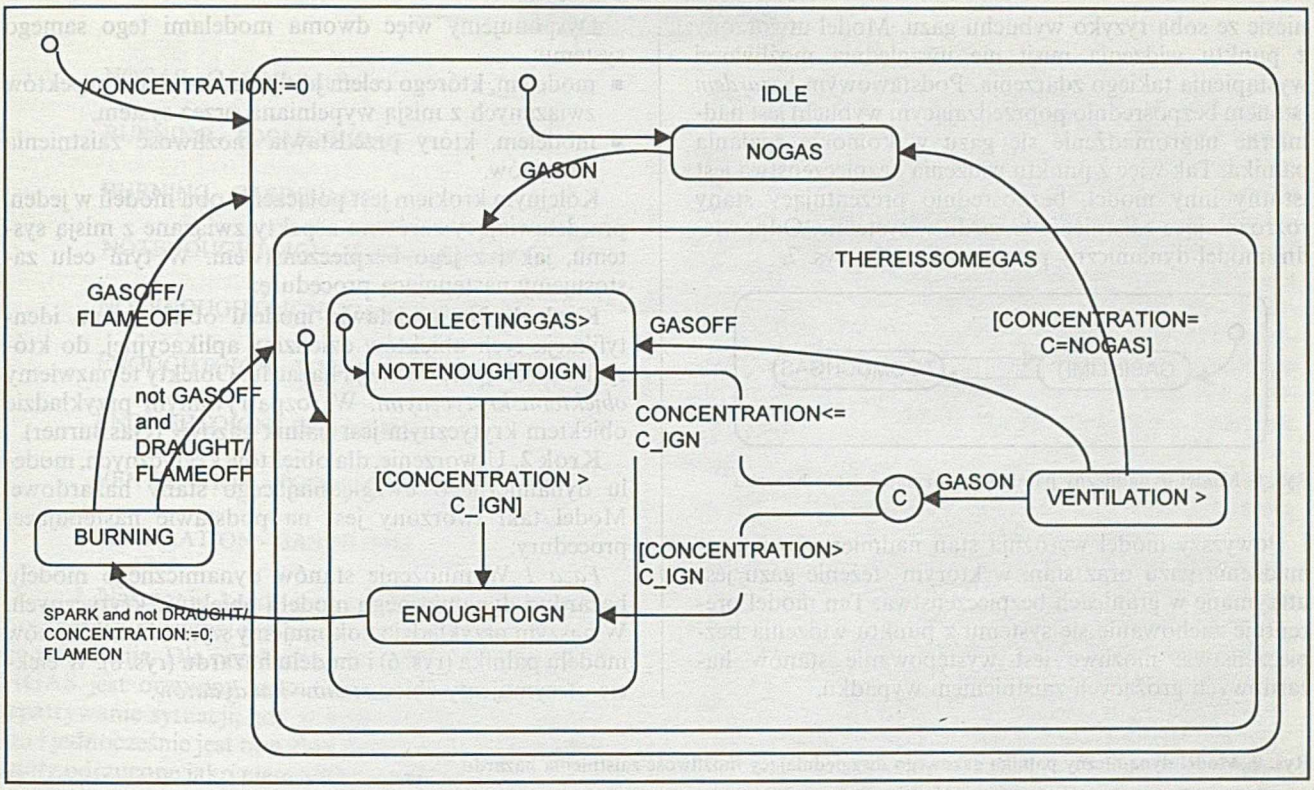


Rys. 5. Model dynamiczny zaworu gazu

Początkowo system sterowania znajduje się w stanie jałowym (IDLE). Generowane przez operatora zdarzenie – polecenie włączenia grzania (HEATINGON) – powoduje otworzenie zaworu gazu (SYGGASON) oraz przejście do stanu przygotowania do zapłonu (PREPARETOIGN) – zapalniczka musi się naładować, a stężenie gazu osiągnąć odpowiedną wartość. Po upłynięciu odpowiedniego czasu ($MAX(C\_IGN, C\_TI) + C\_TS$) system sterujący wysyła do zapalniczki rozkaz generacji iskry (SYGIGNON) oraz przechodzi do stanu oczekiwania na roz-



Rys. 4. Model dynamiczny czujnika temperatury



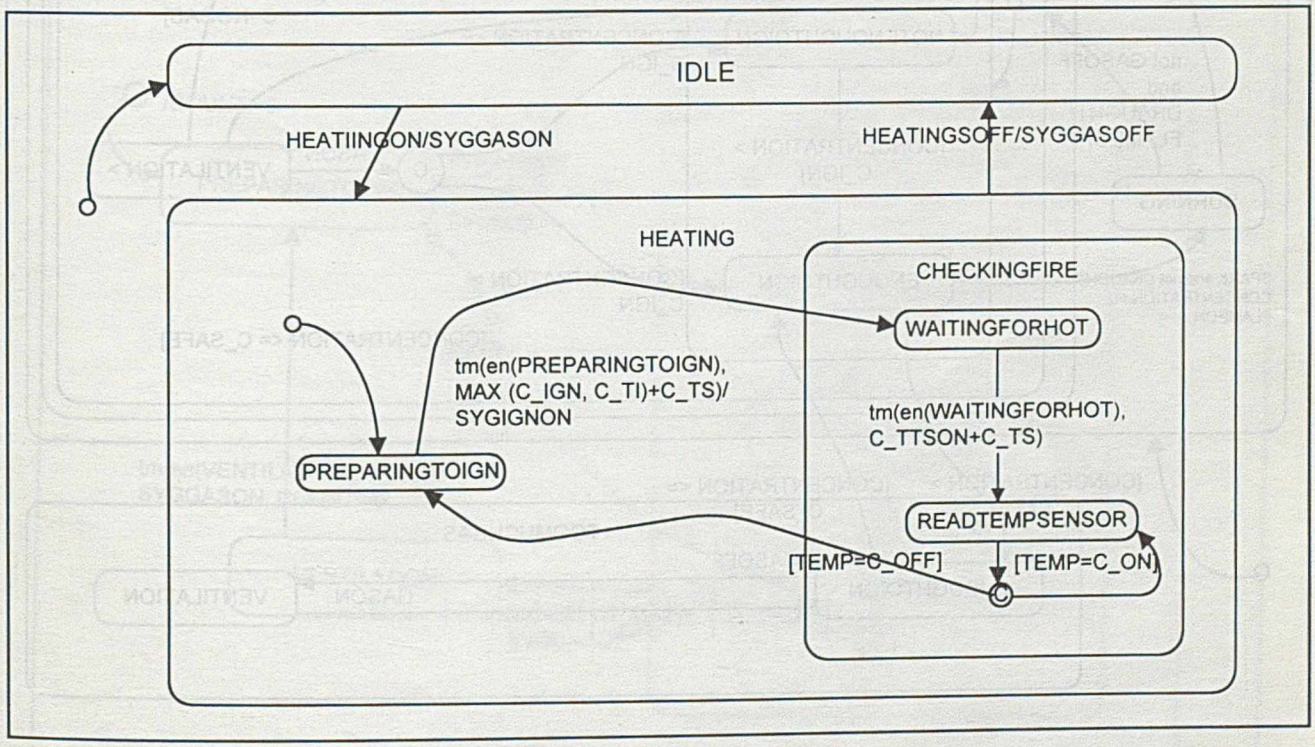
Rys. 6. Model dynamiczny palnika

grzanie czujnika. Po upływie czasu grzania czujnika (*C_TTSON*) system sterujący cyklicznie próbkuje wskazania czujnika. Jeżeli czujnik poinformuje o braku ciepła, system przechodzi do stanu przygotowania do zapłonu i powtarza cykl. Polecenie wyłączenia gazu (HEATINGOFF) powoduje przejście do stanu jałowego (IDLE).

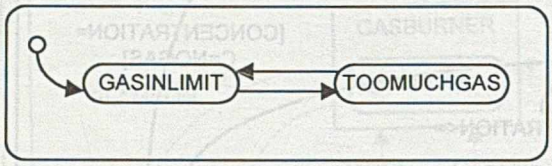
Analiza bezpieczeństwa systemu

Przedstawiony model systemu palnika jest ukierunkowany na opis jego misji związanej z okresowym włączaniem i wyłączaniem płomienia. Jednakże zastosowanie palnika

Rys. 7. Model dynamiczny układu sterowania



niesie ze sobą ryzyko wybuchu gazu. Model utworzony z punktu widzenia misji nie uwzględnia możliwości wystąpienia takiego zdarzenia. Podstawowym *hazardem* (stanem bezpośrednio poprzedzającym wybuch) jest nadmierne nagromadzenie się gazu w komorze spalania palnika. Tak więc z punktu widzenia bezpieczeństwa jest istotny inny model, bezpośrednio prezentujący stany rozróżniające zaistnienie hazardu w systemie. Odpowiedni model dynamiczny przedstawiono na rys. 7.



Rys. 8. Model dynamiczny palnika rozróżniający stan hazardu

Powyższy model wyróżnia stan nadmiernego nagromadzenia gazu oraz stan, w którym stężenie gazu jest utrzymane w granicach bezpieczeństwa. Ten model prezentuje zachowanie się systemu z punktu widzenia bezpieczeństwa: możliwe jest występowanie stanów hazardowych grożących zaistnieniem wypadku.

Dysponujemy więc dwoma modelami tego samego systemu:

- modelem, którego celem jest przedstawienie aspektów związanych z misją wypełnianą przez system,
- modelem, który przedstawia możliwość zaistnienia hazardów.

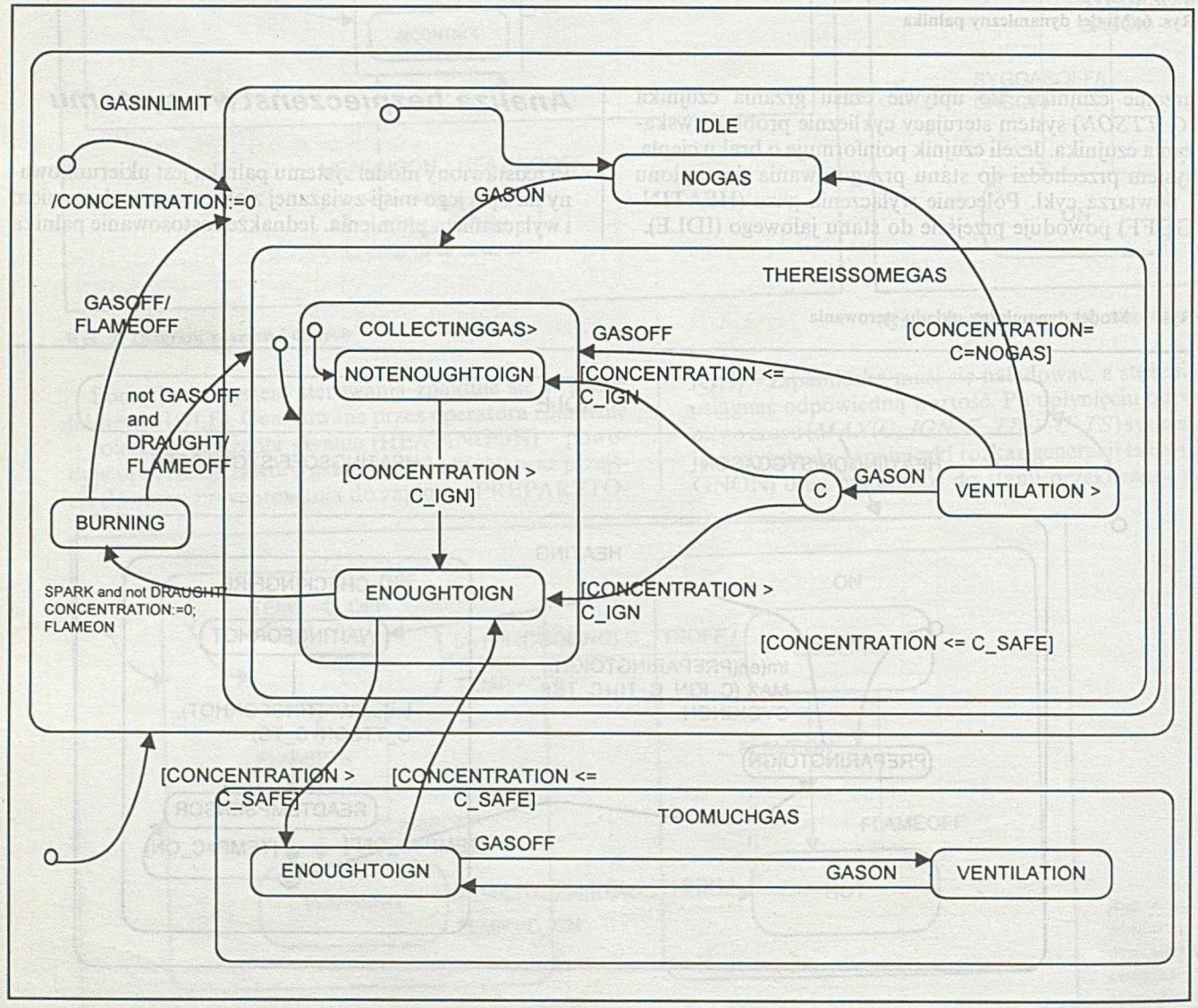
Kolejnym krokiem jest połączenie obu modeli w jeden, przedstawiający zarówno aspekty związane z misją systemu, jak i z jego bezpieczeństwem. W tym celu zastosujemy następującą procedurę:

Krok 1. Na podstawie modelu obiektowego identyfikacja tych obiektów dziedziny aplikacyjnej, do których odwołuje się definicja hazardu. Obiekty te nazwiemy *obiektami krytycznymi*. W rozpatrywanym przykładzie obiektem krytycznym jest palnik gazowy (GasBurner).

Krok 2. Utworzenie, dla obiektów krytycznych, modelu dynamicznego uwzględniającego stany hazardowe. Model taki tworzony jest na podstawie następującej procedury:

Faza 1 Wymnożenie stanów dynamicznego modelu hazardu i dynamicznego modelu obiektów krytycznych. W naszym przykładzie dokonujemy wymnożenia stanów modelu palnika (rys. 6) i modelu hazardu (rys. 8). W efekcie otrzymujemy zbiór *stanów-kandydatów:*

Rys. 9. Model dynamiczny palnika gazowego uwzględniający możliwość zaistnienia hazardu



NOGAS - GASINLIMIT

~~BURNING - TOOMUCHGAS~~

BURNING - GASINLIMIT

~~NOTENOUGHTOIGN - TOOMUCHGAS~~

NOTENOUGHTOIGN - GASINLIMIT

ENOUGHTOIGN - TOOMUCHGAS

ENOUGHTOIGN - GASINLIMIT

VENTILATION - TOOMUCHGAS

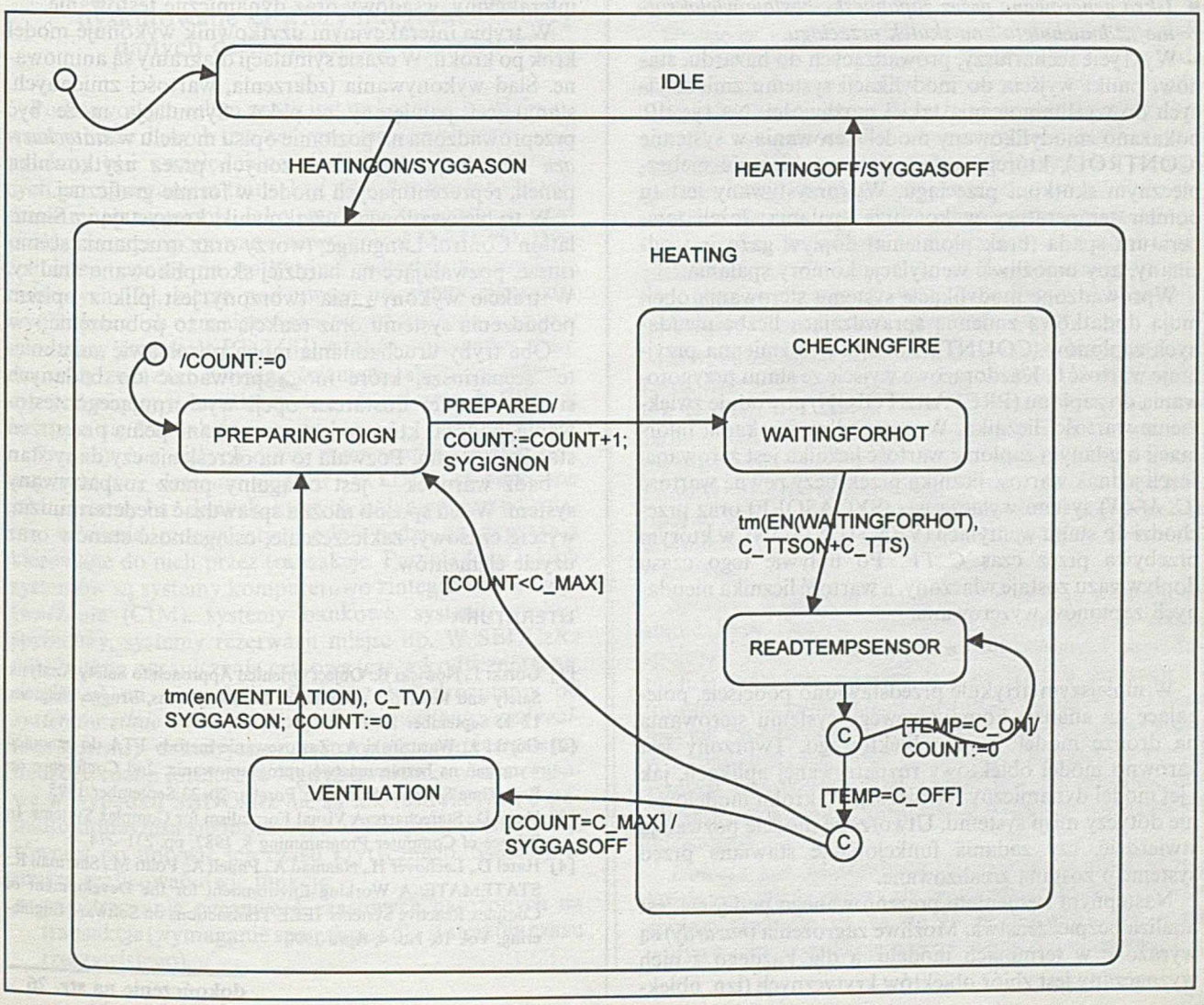
VENTILATION - GASINLIMIT

Faza 2 Następnie, każdy stan-kandydat jest poddawany ocenie walidacyjnej mającej na celu określenie szansy jego zaistnienia. Dla przykładu, stan NOGAS-TOOMUCHGAS jest oceniany jako niemożliwy (nie ma sensu rozpatrywanie sytuacji, gdy w komorze spalania nie ma gazu i jednocześnie jest tam zbyt dużo gazu). Stany, które zostały odrzucone jako niemożliwe, zostały przekreślone.

Faza 3 W efekcie fazy 2. eliminowane są niektóre ze stanów-kandydatów, odnoszące się do zaistnienia hazardu w ramach pewnych stanów obiektu krytycznego (np. hazard nie jest możliwy w stanie BURNING – spalanie usuwa gaz z komory palnika). Pozostałe stany obiektu krytycznego zostają „rozbite" na część *bezpieczną* i część *niebezpieczną*, przez dopuszczenie możliwości zaistnienia hazardu (dotyczy to stanów ENOUGHTOIGN i VENTILATION). Powstały w ten sposób zbiór stanów stanowi punkt wyjścia przy tworzeniu modelu dynamicznego obiektu krytycznego, z uwzględnieniem możliwości zaistnienia hazardu. Przejścia w tym modelu są tworzone następująco:

- zachowane są przejścia modelu obiektu krytycznego, z tym że dla stanów podlegających rozbiciu, przejścia te prowadzą do części bezpiecznej stanu,
- dla każdego stanu podlegającego rozbiciu, definiowane jest przejście pomiędzy częścią bezpieczną i częścią niebezpieczną. Definicja tego przejścia wymaga oceny walidacyjnej, uwzględniającej specyfikę rozpatywanej aplikacji. W naszym przykładzie wystąpi przejście pomiędzy stanami ENOUGHTOIGN-GASINLIMIT oraz ENOUGHTOIGN-TOOMUCHGAS (stężenie gazu narasta i najpierw jest go wystarczająco dużo, aby spowodować normalny zapłon, a dopiero

Rys. 10. Model sterowania eliminujący niebezpieczne skutki przeciągu



potem stężenie staje się niebezpieczne) oraz od stanu VENTILATION-TOOMUCHGAS do stanu VENTILATION-GASINLIMIT (podczas wentylacji, stężenie gazu maleje).
- rozpatrywane są możliwe przejścia pomiędzy stanami niebezpiecznymi, związane z odpowiednikami tych przejść w części bezpiecznej modelu.

Krok 3. W efekcie powyższych zabiegów powstaje model dynamiczny obiektu krytycznego, jawnie wyróżniający możliwość zaistnienia hazardu. Dla palnika gazowego model taki przedstawia rys. 9.

Nowo utworzony model jest poddawany, łącznie z pozostałymi modelami obiektów systemu, analizie mającej na celu stwierdzenie czy rozpatrywany system jest bezpieczny (tzn. czy stan hazardu jest osiągalny). Analiza jest realizowana automatycznie przez narzędzie STATEMATE na podstawie badania osiągalności stanów. Analiza przeprowadzona w stosunku do systemu sterowania palnikiem gazowym prowadzi do wykrycia dwóch scenariuszy zdarzeń prowadzących do hazardu:
- *Operator wielokrotnie, w krótkich odstępach czasu, włącza i wyłącza grzanie. Na skutek bezwładności zapalniczki nie dochodzi do zapłonu, natomiast następuje gromadzenie się gazu w komorze spalania (każdorazowo system sterujący wykonuje procedurę przygotowania palnika (ładowanie zapalniczki, zbieranie gazu) jednak operator wyłącza palnik, zanim dojdzie do zapłonu);*
- *Iskra generowana przez zapalniczkę zostaje wielokrotnie „zdmuchnięta" na skutek przeciągu.*

Wykrycie scenariuszy, prowadzących do hazardu, stanowi punkt wyjścia do modyfikacji systemu zmierzających do wyeliminowania takiej możliwości. Na rys. 10. pokazano zmodyfikowany model sterowania w systemie (CONTROL), którego celem jest zapobieżenie niebezpiecznym skutkom przeciągu. Wykorzystywany jest tu pomiar temperatury w komorze spalania. Jeżeli temperatura spada (brak płomienia) dopływ gazu jest odcinany, aby umożliwić wentylację komory spalania.

Wprowadzone modyfikacje systemu sterowania obejmują dodatkową zmienną sprawdzającą liczbę nieudanych zapłonów (COUNT). Początkowo zmienna przyjmuje wartość 0. Każdorazowe wyjście ze stanu przygotowania do zapłonu (PREPARETOIGN) powoduje zwiększenie wartości licznika. W przypadku uzyskania informacji o udanym zapłonie wartość licznika jest zerowana. Jeżeli jednak wartość licznika przekroczy pewną wartość (*C_MAX*) system wyłącza gaz (SYGASOFF) oraz przechodzi do stanu wentylacji (VENTILATION), w którym przebywa przez czas *C_TV*. Po upływie tego czasu dopływ gazu zostaje włączony, a wartość licznika nieudanych zapłonów wyzerowana.

* * *

W niniejszym artykule przedstawiono podejście, polegające na analizie komputerowego systemu sterowania na drodze modelowania obiektowego. Tworzony jest zarówno model obiektowy rozpatrywanej aplikacji, jak i jej model dynamiczny. W pierwszym kroku modelowanie dotyczy misji systemu. Utworzone modele pozwalają stwierdzić, czy zadania funkcjonalne stawiane przed systemem zostaną zrealizowane.

Następnym elementem prezentowanego podejścia jest analiza bezpieczeństwa. Możliwe zagrożenia (*hazardy*) są wyrażone w terminach modelu, a dla każdego z nich wyznaczany jest zbiór obiektów krytycznych (tzn. obiektów, bezpośrednio uczestniczących w tym hazardzie). Następnie jest tworzony model obiektów krytycznych z jawnym uwzględnieniem stanów niebezpiecznych. Służy on do analizy bezpieczeństwa mającej na celu wykrycie scenariuszy zdarzeń prowadzących do hazardu. Analiza ta jest prowadzona przez badanie osiągalności stanów niebezpiecznych.

Wykryte scenariusze hazardów stanowią punkt wyjścia do modyfikacji systemu, których celem jest uniemożliwienie ich zaistnienia. Dzięki temu, przez celowe zmiany wprowadzane do systemu, jest zwiększane jego bezpieczeństwo.

Podejście przedstawione w niniejszym artykule polega na „całościowym" modelowaniu rozpatrywanego systemu i analizie własności powstałego w ten sposób modelu. Podejście to jest komplementarne do analiz opartych na modelach „częściowych", takich jak – na przykład – analiza drzew błędów (ang. *Fault Tree Analysis*) rozpatrywana w [7]. W [8] przedstawiono użycie podejścia obiektowego do modelowania błędów. Kombinacja wyników analiz podejmowanych z różnych punktów wyjściowych prowadzi do wzmocnienia argumentacji na rzecz bezpieczeństwa i w efekcie umożliwia zapanowanie nad ryzykiem związanym z danym zastosowaniem.

W prowadzonych badaniach, do analizy osiągalności stanów oraz do określenia scenariuszy hazardów wykorzystywano narzędzie STATEMATE. Przewiduje ono trzy podstawowe tryby analizy modelu dynamicznego: interakcyjny, wsadowy oraz dynamiczne testowanie.

W trybie interakcyjnym użytkownik wykonuje model krok po kroku. W czasie symulacji diagramy są animowane. Ślad wykonywania (zdarzenia, wartości zmiennych, stany) jest pamiętany w pliku. Symulacja może być przeprowadzona na poziomie opisu modelu w *statechartach* lub na poziomie tworzonych przez użytkownika paneli, reprezentujących model w formie graficznej.

W trybie wsadowym użytkownik, korzystając z Simulation Control Language, tworzy oraz uruchamia scenariusze, pozwalające na bardziej skomplikowane analizy. W trakcie wykonywania tworzony jest plik z opisem pobudzenia systemu oraz reakcją na to pobudzenie.

Oba tryby uruchamiania mogą pozostawić „nietknięte" scenariusze, które mogą prowadzić do badanych stanów. Pakiet dostarcza opcji wyczerpującego testowania, podczas którego jest generowana pełna przestrzeń stanów systemu. Pozwala to na określenie czy dany stan – bądź warunek – jest osiągalny przez rozpatrywany system. W ten sposób można sprawdzać niedeterminizm, wyścig czasowy, zakleszczenie, osiągalność stanów oraz użycie elementów.

LITERATURA

[1] Górski J., Nowicki B.: Object Oriented Approach to Safety Analysis, Safety and Reliabilty of Software Based Systems, Brugge, Belgium, 12–15 September 1995

[2] Górski J., Wardziński A.: Zastosowanie metody FTA do generacji wymagań na bezpieczeństwo oprogramowania. 2nd Conference on Real Time Systems, Szklarska Poręba, 20–23 September 1995

[3] Harel D.: Starecharts: A Visual Formalism for Complex Systems. In Science of Computer Programming 8, 1987, pp. 231–274

[4] Harel D., Lachover H., Naamad A., Pnueli A., Politi M., Sherman R.: STATEMATE: A Working Environment for the Development of Complex Reactive Systems. IEEE Transactions on Software Engineering, Vol. 16, No. 4, April 1990

dokończenie na str. 26

Więzy integralności a zarządzanie transakcjami w SQL-owych systemach baz danych czasu rzeczywistego

Tadeusz Pankowski
Katedra Automatyki, Robotyki i Informatyki
Politechnika Poznańska

W systemach baz danych czasu rzeczywistego (SBDCzRz) specjalne znaczenie mają dwa problemy, związane z przetwarzaniem transakcji: *spełnienie ograniczeń czasowych nałożonych na transakcje* (problem specyficzny dla wszystkich systemów czasu rzeczywistego) i *zapewnienie integralności bazy danych* (problem specyficzny dla wszystkich systemów baz danych). W pracy rozważane są obydwa te zagadnienia – omawiane są protokoły zarządzania transakcjami współbieżnymi z uwzględnieniem ograniczeń czasowych oraz dyskutowane są więzy integralności bazy danych definiowanych w języku SQL.

Podstawowym celem pracy jest wykazanie, że zdefiniowane w bazie danych tzw. *referencyjne więzy integralności* powinny być uwzględniane w algorytmach zarządzających przetwarzaniem transakcji współbieżnych z ograniczeniami czasowymi. Jako metodę rozwiązania tego problemu zaproponowano nowy protokół RIPTO. Protokół ten, oparty na tradycyjnej metodzie znaczników czasowych [2], bierze pod uwagę priorytety transakcji wynikające z ich ograniczeń czasowych (metoda zaproponowana w protokole PTO [8]), a także wprowadza nowe elementy związane z uwzględnieniem zależności referencyjnych między jednostkami danych, o dostęp do których ubiegają się transakcje.

Systemy baz danych czasu rzeczywistego (SBDCzRz, ang. *real-time database systems* – RTDBS) są określane jako systemy baz danych, których zadaniem jest udzielanie w czasie rzeczywistym odpowiedzi na zapytania kierowane do nich przez transakcje. Przykładami takich systemów są systemy komputerowo zintegrowanego wytwarzania (CIM), systemy bankowe, systemy obsługi sprzedaży, systemy rezerwacji miejsc itp. W SBDCzRz nałożone są ograniczenia czasowe (czasy krytyczne) (ang. *deadlines*) na transakcje. Systemy te odróżniamy od *systemów silnie uwarunkowanych czasowo* (ang. *hard real-time systems*) – takich jak np. systemy sterowania, systemy nadzoru ruchu lotniczego – gdyż uwarunkowania czasowe w wypadku SBDCzRz nie są tak restrykcyjne, a ponadto dopuszcza się możliwość wykonywania transakcji również po upływie ich czasu krytycznego. Systemy te muszą realizować dwie funkcje:

- przestrzeganie ograniczeń czasowych nałożonych na transakcje (wymaganie specyficzne dla systemów czasu rzeczywistego),
- zachowanie integralności bazy danych, również w wypadku awarii (wymaganie wspólne dla wszystkich systemów baz danych).

Pierwsze z powyższych wymagań stanowi istotnie odmienną cechę SBDCzRz odróżniającą te systemy od konwencjonalnych systemów baz danych. Stąd głównym obszarem zainteresowań badawczych jest poszukiwanie takich protokołów planowania wykonywania transakcji, aby było możliwe sprostanie wymaganiom czasowym nałożonym na transakcje. W tym celu opracowanych zostało wiele różnych protokołów zarządzania transakcjami współbieżnymi i uwarunkowanymi czasowo. Wiele z nich określa priorytety transakcji wynikające z ich ograniczeń czasowych.

Drugie ze wspomnianych wymagań dotyczy integralności bazy danych, a więc jej niesprzeczności i wiarygodności dla użytkownika. Zapewnione jest to wówczas, gdy w każdym stanie bazy danych spełnione są wszystkie warunki niesprzeczności zwane *więzami integralności bazy danych* [3, 4, 5, 6]. Taki stan nazywamy stanem spójnym. Więzy te są specyfikowane w języku definiowania bazy danych (np. SQL) i zapamiętane w katalogu systemowym. Transakcje działające w bazie danych mają właściwość ACID, tzn. mają właściwości: atomowości (ang. *atomicity*), spójności (ang. *consistency*), izolacji (ang. *isolation*) i trwałości (ang. *durability*) [2, 3].

Atomowość oznacza, że transakcja traktowana jest jako jedna jednostka operacyjna. Zatem efekt jej wykonania jest efektem wykonania albo wszystkich jej akcji, albo żadnej. Jeśli więc wykonanie transakcji zostanie awaryjnie przerwane (błędy w danych, naruszenie więzów integralności, awaria komunikacji w sieci, awaria zasilania itp.), to system jest odpowiedzialny za to, aby wszelkie dokonane przez nią zmiany w bazie danych zostały wycofane.

Spójność transakcji oznacza, że działając w spójnym stanie bazy danych pozostawia ją również w stanie spójnym. Ponieważ jednak warunki spójności bazy danych, tj. więzy integralności, są zewnętrzne w stosunku do transakcji i są pamiętane w katalogu systemowym, zatem nad ich zachowaniem czuwa specjalny podsystem, który w wypadku stwierdzenia ich naruszenia generuje sygnał sytuacji wyjątkowej. Jeśli sygnał ten nie zostanie obsłużony przez samą transakcję, to następuje awaryjne jej odrzucenie.

Izolacja jest właściwością transakcji, w myśl której każda transakcja przez cały czas widzi bazę danych jako spójną. Oznacza to, że na to postrzeganie nie mają wpływu inne transakcje wykonywane współbieżnie.

Trwałość oznacza, że jeśli transakcja została zatwierdzona, to wywołany przez nią efekt pozostaje utrwalony w bazie danych i żadna następująca po zatwierdzeniu awaria nie może tego efektu unieważnić.

Z właściwości tych wynika, że każda transakcja przekształca spójny stan bazy danych w stan spójny, chociaż w trakcie działania transakcji spójność nie musi być zachowana. System czuwający nad zachowaniem więzów integralności będzie więc powodował odrzucenie tych transakcji, które naruszyłyby te więzy. Ukazuje to związek między wykonywaniem transakcji a kontrolowaniem więzów integralności. W dalszym ciągu pokażemy, że można te dwa zagadnienia w pewnym stopniu połączyć wykorzystując definicje więzów integralności w nowym protokole zarządzania transakcjami w bazie danych.

Zarządzanie transakcjami

W systemach baz danych czasu rzeczywistego przyjmujemy, że z każdą transakcją jest związany jej czas krytyczny (ang. *deadline*), a niekiedy także lista elementów danych, o dostęp do których transakcja się ubiega. Zarządca transakcji przypisuje transakcjom identyfikatory oraz priorytety czasu rzeczywistego wynikające z ich czasów krytycznych. Najczęściej przyjmowana jest strategia EDF (ang. *Earliest Deadline First*), oznaczająca, że transakcja z wcześniejszym czasem krytycznym ma wyższy priorytet. Przyjmuje się ponadto, że ograniczenia czasowe są miękkie, tzn. każda transakcja jest wykonywana nawet po upływie jej czasu krytycznego. Współbieżne ubieganie się o dostęp do danych jest kontrolowane przez planistę. Planista przydziela prawa dostępu do danych na podstawie wykonywanego *protokołu sterowania transakcjami współbieżnymi*.

Wśród protokołów sterowania transakcjami współbieżnymi w bazach danych możemy wyróżnić:

Protokoły oparte na blokowaniu danych. Każda transakcja musi uzyskać blokadę do odczytu dla każdej odczytywanej jednostki danych (*blokada współdzielenia*) i blokadę do zapisu dla każdej zapisywanej jednostki danych (*blokada wyłączności*). Dwa żądania blokad są w *konflikcie*, jeśli dotyczą tej samej jednostki danych i co najmniej jedna z nich jest blokadą wyłączności. Konfliktowe żądania blokad mogą prowadzić do opóźnień w wykonaniu transakcji, a niekiedy także do zakleszczeń. Do protokołów tej klasy zaliczamy: protokoły oparte na blokowaniu dwufazowym bez uwzględniania priorytetów [2], protokół z dziedziczeniem priorytetów [7], protokół blokowania uwzględniający priorytety transakcji [1], protokół oparty na podwyższaniu priorytetów [7], protokół oparty na priorytecie danych [8]. Ważną cechą dwóch ostatnich z tych protokołów jest to, że zakładają one, iż każda transakcja podaje wykaz jednostek danych, do których będzie chciała uzyskać dostęp. Wymaganie to w wielu wypadkach jest możliwe do zrealizowania, na przykład w systemie sprzedaży z góry wiadomo, że konieczne będzie odwoływanie się do stanu i ceny towarów, a w systemie bankowym – do stanu rachunku klienta. Każdej jednostce danych przypisywany jest najwyższy priorytet spośród priorytetów transakcji deklarujących dostęp do tej jednostki danych. Priorytet ten jest następnie uwzględniany przy udostępnianiu blokad dla poszczególnych transakcji.

Protokoły oparte na uporządkowaniu według znaczników czasowych. Cechą charakterystyczną tych protokołów jest to, że nie dopuszczają do zakleszczeń. W wypadku więc systemów rozproszonych nie ma konieczności tworzenia procesów koordynujących wykrywanie globalnych zakleszczeń. Protokół oparty na priorytetach i znacznikach czasowych PTO (ang. – *priority-based timestamp-ordering protocol*) [8] zakłada ponadto, że transakcje łączone są w grupy wyznaczone przez przedziały znaczników czasowych o ustalonej długości Δ. Każdej transakcji przypisywany jest tak zwany *grupowy znacznik czasowy* będący funkcją znacznika czasowego grupy, do której transakcja należy, i *znacznik czasu rzeczywistego* będący funkcją priorytetu tej transakcji. Ten dwupoziomowy znacznik czasowy dziedziczy jednostka danych od tej transakcji, która jako ostatnia miała do niej dostęp.

Protokoły optymistyczne. Protokoły tej klasy stosują „regułę 50%": jeśli co najmniej połowa transakcji będących w konflikcie z transakcją zatwierdzaną ma wyższy od niej priorytet, to transakcja ta jest opóźniana do momentu zakończenia wszystkich transakcji o wyższym priorytecie; w przeciwnym wypadku zezwala się transakcji na zakończenie, a transakcje będące z nią w konflikcie zostają odrzucone.

Niektóre z przytoczonych powyżej protokołów wymagają wcześniejszego określenia zbioru jednostek danych, o dostęp do których ubiega się transakcja. Każdej jednostce danych jest przypisywany najwyższy priorytet spośród priorytetów wszystkich ubiegających się o nią transakcji. Żaden z tych protokołów nie bierze jednak pod uwagę faktu, że w tak określonym zbiorze jednostek danych mogą występować pewne zależności, które mogą mieć wpływ na kolejność wykonywania transakcji. W wypadku SQL-owych baz danych klasę takich zależności wyznaczają *referencyjne więzy integralności*. Ilustruje to następujący przykład:

PRZYKŁAD 1. Niech w systemie SQL-owej bazy danych istnieją dwie tablice:

```
DOSTAWCA (NrDost, ...)
DOSTAWA (NrDost, NrTow, ...)
```

o następujących definicjach:

```
CREATE TABLE DOSTAWA
(NrDost CHAR (5) NOT NULL,
...
PRIMARY KEY (NrDost)),
```

oraz

```
CREATE TABLE DOSTAWA
(NrDost CHAR (5),
NrTow CHAR(10),
...
FOREIGN KEY (NrDost) REFERENCES DOSTAWCA ON DELETE CASCADE).
```

Fraza FOREIGN KEY w tablicy DOSTAWA definiuje zależność referencyjną, na mocy której każda niezerowa wartość w kolumnie DOSTAWA. NrDost musi wystąpić w kolumnie DOSTAWCA.NrDost. Wprowadza więc zależność typu podrzędny-nadrzędny między tymi jednostkami danych (kolumnami). W szczególności, jeśli wprowadzany jest nowy wiersz do tablicy DOSTAWA, to transakcja go wprowadzająca zostanie odrzucona, jeśli wartość pola NrDost nie występuje w kolumnie DOSTAWA.NrDost.

Rozważmy transakcje realizujące operacje INSERT:

T_1: INSERT$_1$ [DOSTAWA.NrDost, v],
T_2: INSERT$_2$ [DOSTAWCA.NrDost, v].

Transakcja T_1 próbuje wpisać nową wartość v do kolumny DOSTAWA.NrDost. System realizuje również operację odczytu DOSTAWCA.NrDost, aby sprawdzić, czy v występuje w tej kolumnie. Jeśli nie występuje, to transakcja T_1 zostanie odrzucona. Stąd wniosek, że dla operacji typu INSERT, niezależnie od priorytetów transakcji, należałoby najpierw wykonać transakcję T_2, a więc transakcję na tablicy nadrzędnej. Odwrotna sytuacja miałaby miejsce w wypadku operacji UPDATE i DELETE – wówczas niezależnie od priorytetów należałoby najpierw wykonać operacje na tablicy podrzędnej.

Powyższy przykład uzasadnia potrzebę tworzenia takich protokołów zarządzania transakcjami współbieżnymi w SBDCzRz, które uwzględniałyby semantykę danych, o dostęp do których ubiegają się transakcje. Przykładem takiej właściwości semantycznej są zależności referencyjne definiowane w języku SQL.

Więzy integralności w języku SQL

W SQL wyróżnia się trzy typy więzów integralności:

- więzy jednoznaczności i określoności (ang. *unique constraints, notnull constraints*),
- więzy referencyjne (ang. *referential constraints*),
- więzy kontroli (ang. *check constraints*).

Sprawdzanie więzów integralności zależy od ustalenia trybu tego sprawdzania w bieżącej transakcji. Jeśli tryb jest *natychmiastowy* (ang. *immediate*) więzy są sprawdzane na końcu każdego wyrażenia SQL. Jeśli tryb jest opóźniony (ang. *deferred*) to sprawdzanie odbywa się na końcu transakcji lub w odpowiedzi na specjalne polecenie. Domyślnym trybem jest tryb natychmiastowy. Jeśli zostanie stwierdzone naruszenie więzu integralności, to jest generowany warunek *wyjątku* (ang. *exception condition*).

Więzy jednoznaczności służą do zdefiniowania w tablicy bazy danych *klucza głównego* (co najwyżej jednego dla każdej tablicy) i *kluczy alternatywnych* (w jednej tablicy może być ich kilka). Kolumny, tworzące klucz główny, nie mogą zawierać wartości zerowych (nieokreślonych). Klucze główne są definiowane za pomocą słowa kluczowego PRIMARY KEY, a klucze alternatywne za pomocą słowa kluczowego UNIQUE. Wszystkie klucze w tablicy mogą być różne, ale nie muszą być minimalne. Kolumny, tworzące klucz główny, nie mogą zawierać wartości zerowych (konieczne słowo kluczowe NOT NULL w definicji kolumn), natomiast mogą je zawierać kolumny, które tworzą klucz alternatywny. Warunek klucza alternatywnego jest więc spełniony wtedy, gdy żadne z dwóch różnych wierszy w tablicy nie mają takich samych niezerowych wartości dla tego klucza.

Definicja **więzu referencyjnego** zawarta w definicji tablicy R określa *klucz obcy* K w R i wiąże go z kluczem L w tablicy U; klucz L nazywany jest *kluczem wskazywanym* (ang. *referenced key*), tablica R *tablicą wskazującą* (ang. *referencing table*), a tablica U – *tablicą wskazywaną* (ang. *referenced table*). Klucz wskazywany może być kluczem głównym lub jednym z kluczy alternatywnych tablicy U. Jeśli L jest pominięty, to domyślnie przyjmuje się, że chodzi o klucz główny tablicy U. Mówimy, że jest określona wówczas zależność referencyjna między $R.K$ i $U.L$, co zapisujemy: $R.K \xrightarrow[ref]{} U.L$, i mówimy, że $R.K$ jest referencyjnie zależne od $U.L$. Każdy referencyjny więz integralności ma podany (jawnie lub domyślnie) *typ dopasowania* (ang. *match type*), określający warunki spełnienia danego więzu referencyjnego. W zależności od typu dopasowania wyróżniamy trzy wypadki *spełniania* więzów referencyjnych. Niech $R.K \xrightarrow[ref]{} U.L$ oznacza więz referencyjny, gdzie R, K, U, L mają znaczenie opisane powyżej, wówczas:

1) jeśli typ dopasowania jest pominięty, to $R.K \xrightarrow[ref]{} U.L$ jest spełniony wtedy i tylko wtedy, gdy dla każdego wiersza $r \in R$ albo pewna wartość w $r[K]$ jest wartością zerową, albo istnieje wiersz $u \in U$ taki, że $r[K] = u[L]$;
2) jeśli typ dopasowania jest PARTIAL, to $R.K \xrightarrow[ref]{} U.L$ jest spełniony wtedy i tylko wtedy, gdy dla każdego wiersza $r \in R$ albo wszystkie wartości w $r[K]$ są wartościami zerowymi, albo jedna z wartości w $r[K]$ jest niezerowa i istnieje wiersz $u \in U$ taki, że $r[K] = u[L]$;
3) jeśli typ dopasowania jest FULL, to $R.K \xrightarrow[ref]{} U.L$ jest spełniony wtedy i tylko wtedy, gdy dla każdego wiersza $r \in R$ albo wszystkie wartości w $r[K]$ są wartościami zerowymi, albo żadna z wartości w $r[K]$ nie jest zerowa i istnieje wiersz $u \in U$ taki, że $r[K] = u[L]$.

Wiersze r i u występujące w powyższych warunkach nazywamy *wierszami dopasowanymi*. W tabeli 1 pokazano, kiedy są spełnione więzy referencyjne dla trzech typów dopasowania i dla trzech wypadków wartości klucza:

a) wszystkie składowe wartości klucza są zerowe,
b) istnieje co najmniej jedna składowa zerowa i co najmniej jedna niezerowa,
c) wszystkie składowe są niezerowe.

W definicji referencyjnych więzów integralności może być zawarta *reguła usuwania* (ang. *delete rule*) i *reguła aktualizacji* (ang. *update rule*). Reguły te mogą być typu CASCADE, SET NULL lub SET DEFAULT i definiują działanie transakcji w wypadku operacji związanych z usuwaniem lub aktualizacją bazy danych.

Tabela 1. Określenie dopasowania wierszy $r \in R$ i $u \in U$ względem kluczy K i L

	Wartość klucza R[K]	Typ dopasowania pominięty	Typ dopasowania PARTIAL	Typ dopasowania FULL
(a)	(null,null) ∈ R[K]	TRUE	TRUE	TRUE
(b)	(null,x) ∈ R[K]	TRUE	(null,x) ∈ U[L]	FALSE
(c)	(x,y) ∈ R[K]	(x,y) ∈ U[L]	(x,y) ∈ U[L]	(x,y) ∈ U[L]

W wypadku transakcji usuwania wiersza u z relacji U działanie to jest w poszczególnych wypadkach następujące:

1) jeśli pominięta jest reguła usuwania, to transakcja usunięcia wiersza u z tablicy U jest odrzucona, gdy istnieje wiersz r w R dopasowany do u;
2) jeśli reguła usunięcia jest typu CASCADE, to usunięcia wiersza u z U powoduje usunięcie wszystkich wierszy r z R dopasowanych do u;
3) jeśli reguła usunięcia jest typu SET NULL, to usunięcie wiersza u z U powoduje przypisanie wartości zerowych kluczowi K we wszystkich wierszach r z R dopasowanych do u;
4) jeśli reguła usunięcia jest typu SET DEFAULT, to usunięcie wiersza u z U powoduje przypisanie wartości domyślnych wszystkim kolumnom tworzącym klucz K we wszystkich wierszach r z R dopasowanych do u.

W wypadku aktualizacji semantyka transakcji jest następująca:
Niech T będzie transakcją, zmieniającą wartość w pewnej kolumnie A, należącej do L, w pewnym wierszu u z U. Niech B będzie kolumną należącą do K, odpowiadającą A. Wówczas:

1) jeśli reguła aktualizacji jest pominięta, to T jest odrzucona, jeżeli istnieje wiersz $r \in R$ dopasowany do u;
2) jeśli reguła aktualizacji jest typu CASCADE, to wszystkie wiersze $r \in R$ dopasowane do u, są aktualizowane w taki sposób, aby $r[B]$ pozostało równe z $u[A]$;
3) jeśli reguła aktualizacji jest typu SET NULL, to:
   - we wszystkich wierszach $r \in R$ dopasowanych do u, polu $r[B]$ przypisywana jest wartość zerowa, jeżeli pominięty jest typ dopasowania;
   - we wszystkich wierszach $r \in R$ dopasowanych do u, wszystkim polom w $r[K]$ są przypisywane wartości zerowe, jeżeli typ dopasowania jest FULL;
4. jeśli reguła aktualizacji jest typu SET DEFAULT, to we wszystkich wierszach $r \in R$ dopasowanych do u, polom $r[B]$ jest przypisywana wartość domyślna zdefiniowana dla kolumny B.

Definicja ***więzów kontrolnych*** dotyczy trzech rodzajów ograniczeń semantycznych:

- w definicji kolumny określa warunek, jaki muszą spełniać wartości w tej kolumnie, np.:
  CREATE TABLE T
  (A CHAR (1) CHECK (A = 'Y' or A = 'N'),
  B CHAR (1) CHECK (B LIKE (SELECT B FROM U WHERE C = 'M')))
- w definicji tablicy określa warunek, jaki muszą spełniać wiersze należące do tej tablicy, np.:
  CREATE TABLE T
  (A CHAR (1),
  B CHAR (1),
  CHECK ((A = 'Y' AND B = 'N') OR (A = 'N' AND B = 'Y')))
- w oddzielnej *definicji asercji* określa warunek odnoszący się do kilku tablic, np.:
  CREATE ASSERTION ASERCJA
  CHECK (NOT EXISTS
  SELECT K FROM T
  WHERE K IN (SELECT K FROM U)),
  co oznacza, że kolumna K w tablicy T musi mieć rozłączny zbiór wartości z kolumną K w tablicy U.

Protokół RIPTO

W klasycznym protokole opartym na znacznikach czasowych (protokół TO [2]), decyzje planowania realizacji operacji konfliktowych są oparte na wartościach znaczników czasowych, przypisywanych transakcjom w momencie ich inicjowania. Przy zarządzaniu transakcjami z priorytetami może się jednak zdarzyć, że transakcja T o wysokim priorytecie zostanie odrzucona przy próbie dostępu do jednostki danych, jeśli transakcja T' o niższym priorytecie, ale o późniejszym znaczniku czasowym, uzyskała wcześniej dostęp do tej jednostki danych. Problem ten jest zwany *inwersją priorytetów* (ang. *priority inversion for timestamp-ordering protocols*). W [8] jest proponowany protokół PTO (ang. *priority-based timestamp-ordering protocol*), który przezwycięża problem inwersji priorytetów. Protokół PTO stanowi punkt wyjścia do nowego proponowanego tutaj protokółu RIPTO, który uzwględnia zależności referencyjne (ang. *referential integrity*) między jednostkami danych.

Istotną cechą protokółu PTO jest włączenie priorytetów transakcji wynikających z nałożonych na nie ograniczeń czasowych do przypisywanych transakcjom znaczników czasowych. W tym celu zbiór transakcji jest łączony w grupy w zależności od ich czasu inicjalizacji. Czas jest dzielony na przedziały o pewnej długości Δ, a transakcje, które nadeszły w tym samym przedziale czasu, są umieszczane w tej samej grupie. Każdej transakcji jest przypisywany znacznik dwupoziomowy, utworzony ze znacznika czasowego grupy (*znacznik grupy* – znacznik czasowy odpowiadający czasowi nadejścia pierwszej transakcji należącej do tej grupy) i znacznika czasowego czasu rzeczywistego (znacznika *r-t*), utworzonego na podstawie priorytetów transakcji. Transakcja o najwyższym priorytecie ma najpóźniejszy znacznik *r-t*, tak aby nie została odrzucona przez żadną transakcję z tej samej grupy w wypadku konfliktu dostępu. Tak więc znacznik grupy jest wykorzystywany do planowania dostępu w wypadku transakcji z różnych grup, a znacznik *r-t* – do planowania dostępu w wypadku transakcji należących do tej samej grupy. Każda jednostka danych ma przypisany zarówno znacznik grupy, jak i znacznik *r-t* związany z transakcją, która jako ostatnia miała dostęp do tej jednostki danych. Przypuśćmy, że transakcja T żąda dostępu do jednostki danych D. Jeśli znacznik grupy T jest późniejszy niż znacznik grupy przypisany D, to T uzyskuje dostęp do D i znacznik D jest aktualizowany. Jeśli jest wcześniejszy – T jest odrzucana. W wypadku równych znaczników grup porównywane są znaczniki *r-t*. Odrzucone transakcje otrzymują nowe znaczniki w chwili ich restartowania. Istotnym problemem przy stosowaniu protokółu PTO jest właściwy wybór długości przedziału czasowego Δ.

Zmodyfikujemy obecnie protokół PTO w taki sposób, aby uwzględniał semantykę danych związaną ze zdefiniowanymi w bazie danych zależnościami referencyjnymi. Nowo otrzymany protokół nazwiemy protokółem RIPTO (ang. *referential integrity- and priority-based timestamp-ordering protokol*). Modyfikacja ta dotyczy przede wszystkim sposobu określania znacznika *r-t*. W PTO zależał on jedynie od priorytetu transakcji, natomiast w protokole RIPTO zależy on zarówno od typu transakcji (INSERT, UPDATE lub DELETE), jak i od zależności referencyjnych między danymi, dostępu do których żądają transakcje.

Niech dla tablic R i U określona będzie zależność referencyjna $R.K \xrightarrow[ref]{} U.L$. Niech T_1 i T_2, należące do tej samej grupy czasowej, żądają dostępu do dopasowanych do siebie względem K i L wierszy $r \in R$ i $u \in U$. Wówczas zasada przypisywania znaczników r-t określona jest w tabeli 2, gdzie $zn\_rt(T)$ oznacza znacznik r-t transakcji T, a $zn\_rt(T_2) < zn\_rt(T_1)$ oznacza, że znacznik r-t transakcji T_1 jest późniejszy niż znacznik r-t transakcji T_2. W pozostałych wypadkach obowiązują reguły protokołu PTO [8].

W wypadku operacji typu INSERT uzasadnienie reguły wynika z analizy przykładu 1. Rozważmy wypadek, gdy obydwie transakcje są typu UPDATE. Próba wykonania operacji $UPDATE_1(u[L], v)$ przed operacją $UPDATE_2(r[K], v')$ może mieć następujące skutki:

(a) odrzucenie transakcji T_1, jeśli w definicji zależności referencyjnej pominięto regułę aktualizacji;

(b) kaskadową aktualizację wartości klucza K we wszystkich wierszach tablicy R dopasowanych do u, zgodnie z regułą aktualizacji i typem dopasowania.

Zauważmy, że: (a) odrzucenia transakcji T_1 można by uniknąć, gdyby najpierw wykonać operację $UPDATE_2(r[K], v')$, gdyż po jej zakończeniu może już nie być w R wierszy dopasowanych do u; (b) w drugim wypadku kaskadowa aktualizacja w R może doprowadzić do sytuacji, gdy $UPDATE_2(r[K], v')$ w ogóle nie może być wykonana. Wynika stąd, że transakcja T_2 powinna mieć późniejszy znacznik r-t (wyższy priorytet) niż transakcja T_1. Podobnie można uzasadnić pozostałe zależności podane w tabeli 2.

Uwagi powyższe dotyczą sytuacji, gdy operacje $UPDATE_1(u[L], v)$ i $UPDATE_2(r[K], v')$ są wykonywane w obrębie różnych transakcji. Wykonywanie ich w obrębie jednej transakcji wymaga jedynie określenia opóźnionego trybu sprawdzania spójności bazy danych – wówczas zachowanie więzów integralności nie jest sprawdzane po zakończeniu tych pojedynczych operacji, ale dopiero po zakończeniu całej transakcji. Takie jednak założenie jest mało realistyczne, jeśli przyjąć, że system bazy danych jest systemem pracującym w środowisku wielodostępnym.

Obsługa dostępu do danych odbywa się według następującego algorytmu [8], gdzie zn_gr( ) zwraca znacznik grupy (transakcji i danych), a zn_rt( ) zwraca znacznik r-t:

```
if (zn_gr(T) > zn_gr(D))
    T uzyskuje dostęp do D;
    zn_gr(D) = zn_gr(T);
    zn_rt(D) = zn_rt(T);
else
    if (zn_gr(T) = zn_gr(D) and zn_rt(T) > zn_rt(D))
        T uzyskuje dostęp do D;
        zn_rt(D) = zn_rt(T);
    else
        T jest odrzucana.
```

* * *

W pracy omówiono wpływ więzów integralności definiowanych w SQL-owych systemach baz danych czasu rzeczywistego (SBDCzRz) na zarządzanie transakcjami współbieżnymi w tych systemach. Wśród licznych protokołów zarządzania transakcjami współbieżnymi w SBDCzRz [8] brak protokołów, które uwzględniałyby semantyczne właściwości danych wynikające ze zdefiniowanych więzów integralności. Wydaje się, że uwzględnienie tego rodzaju zależności może mieć wpływ na sprostanie wymaganiom czasowym stawianym transakcjom wykonywanym w takim systemie.

W pracy szczególną uwagę poświęcono zależnościom referencyjnym. Szczegółowo omówiono ich semantykę w relacyjnych bazach danych i sposób ich definiowania w języku SQL. Przedyskutowano także ich wpływ na operacje zmieniające stan bazy danych. Zwrócono uwagę, że w wielu wypadkach uwzględnienie zależności referencyjnych między jednostkami danych może mieć istotny wpływ na planowanie wykonywania transakcji. Na przykład, z góry można przewidzieć, że przy pewnej kolejności wykonywania transakcji niektóre z nich na pewno zostaną odrzucone. Spostrzeżenie to doprowadziło do zaproponowania nowego protokółu zarządzania transakcjami, w którym uwzględnia się zależności referencyjne między jednostkami danych, do których odwołuje się pewna grupa transakcji. Protokół ten, zwany protokołem RIPTO, jest zmodyfikowaną wersją protokołu opartego na znacznikach czasowych TO [2] i protokołu PTO [8]. W [8] wykazano, że protokół PTO może istotnie poprawić wydajność systemu pracującego w czasie rzeczywistym, szczególnie w warunkach dużego obciążenia systemu i przy licznych konfliktach dostępu do danych.

Tabela 2. Reguły przypisywania znaczników r-t transakcjom T_1 i T_2 należącym do tej samej grupy i żądającym wykonania operacji na dopasowanych do siebie wierszach r i u

T_1 / T_2	$INSERT_1(u)$	$UPDATE_1(u[L], v)$	$DELETE_1(u)$
$INSERT_2(r)$	$zn\_rt(T_2) < zn\_rt(T_1)$	$zn\_rt(T_2) < zn\_rt(T_1)$	$zn\_rt(T_2) < zn\_rt(T_1)$
$UPDATE_2(r[K], v')$	$zn\_rt(T_2) < zn\_rt(T_1)$	$zn\_rt(T_2) > zn\_rt(T_1)$	$zn\_rt(T_2) > zn\_rt(T_1)$
$DELETE_2(r)$	$zn\_rt(T_2) > zn\_rt(T_1)$	$zn\_rt(T_2) > zn\_rt(T_1)$	$zn\_rt(T_2) > zn\_rt(T_1)$

Wydaje się, że protokół RIPTO winien się cechować lepszymi efektami, szczególnie wtedy, gdy przetwarzanych ma być współbieżnie w czasie rzeczywistym dużo transakcji, dla których często trzeba sprawdzać zachowanie obowiązujących zależności referencyjnych.

LITERATURA

[1] Abbot R., Garcia-Molina H, Scheduling real-time transactions: a performance evaluation, ACM Transactions on Database Systems, Vol. 17, No. 3, pp. 513–560, 1992

[2] Bernstein P.A., Hadzilacos V., Goodman N.: Concurrency control and recovery in database systems, Addison-Wesley, New York, 1987

[3] Elmasri R., Navathe S.B.: Fundamentals of Database Systems. Second Edition, The Benjamin/Cummings, 1994

[4] Laender A.H.F., Casanova M.A., De Carvalho A.P., Ridolfi L.F.G.G.M.: An analysis of SQL integrity constraints from an entity-relationship model perspective, Information Systems, Vol. 19, No. 4, pp. 331–385, 1994

[5] Markowitz V.M.: Safe referential integrity and null constraint structures in relational databases, Information Systems, Vol. 19, No. 4, pp. 359–378, 1994

[6] Pankowski T.: Podstawy baz danych, Wydawnictwo Naukowe PWN, Warszawa 1992

[7] Sha L., Rajkumar R., Lechoczky J.P.: Concurrency control for distributed real-time databases, ACM SIGMOD Record 17, pp. 82–98, 1988

[8] Ulusoy O., Belford G.G.: Real-time transaction scheduling in database systems, Information Systems, Vol. 18, No. 8, pp. 559–580, 1993.

Obiektowe podejście do modelowania i analizy...

dokończenie ze s. 20

[5] Harel D., Pnueli A., Schmidt J.P., Sherman R.: On the formal semantics of statecharts. In Proceedings of the 2nd IEEE Symposium on Logic in Computer Science (Ithaca, N.Y., June 22–24). IEEE Press, New York, 1987, pp. 54–64

[6] i-Logix Inc., Burlington, MA. Tech. Rep. The semantics of statecharts, 1991

[7] i-Logix Inc., Burlington, STATEMATE-Technical Documentation

[8] Rumbaugh J., Blaha M., Premerlani W., Eddy F. and Lorensen W.: Object-Oriented Modelling and Design, Prentice Hall Int., 1991

Każdy numer INFORMATYKI można kupić w redakcji. Prowadzimy sprzedaż odręczną i wysyłkową. Wystarczy do nas zatelefonować lub napisać.

Nasz nowy adres:
03-450 Warszawa
ul. Ratuszowa 11, p. 628 i 644
tel., fax: 619-11-61

Prosimy pamiętać o prenumeracie INFORMATYKI na przyszły rok!

Komputerowe wspomaganie w projektowaniu układów regulacji

VxWorks® ***kompleksowy system inżynierii oprogramowania czasu rzeczywistego***

- architektura host-target, połączenie komputera nadrzędnego z komputerem sterującym
- oprogramowanie pracujące w dowolnym środowisku sieciowym
- zastosowania w przemyśle, telekomunikacji, transporcie, multimediach i in.
- narzędzia wspomagające tworzenie oprogramowania (CASE)
- sterowniki przemysłowe z systemem operacyjnym VxWorks realizowane w standardzie VME lub w technologii komputerów jednoukładowych
- przenoszenie aplikacji z innych systemów operacyjnych czasu rzeczywistego

Oferujemy wykonanie opracowań naukowych, projektów i wdrożeń oprogramowania do sterowania, monitorowania i wizualizacji procesów oraz dodatkowo:

- oprogramowanie do obliczeń naukowo-technicznych i wizualizacji graficznej: Matlab, Simulink, Matlab Application Toolboxes, MapleV
- oprogramowanie i karty z procesorami sygnałowymi dSpace
- modułowe komputery firmy PEP Modular Computers, spełniające normę jakości ISO 9001
- kursy i szkolenia z zakresu obsługi i zastosowań wymienionego oprogramowania

Wszelkich informacji udziela autoryzowany przedstawiciel producentów oprogramowania:

ControlSoft

ControlSoft

ul.Piastowska 44, 30-067 Kraków

tel.: (012) 36 32 18 *fax: (012) 36 84 11* *E-mail: info@ControlSoft.Krakow.PL*

0/22/95

WIZCON 5.0 - SYSTEM STEROWANIA I WIZUALIZACJI PROCESÓW TECHNOLOGICZNYCH

Nowoczesne, komputerowe układy automatyki poza urządzeniami zbierającymi i przetwarzającymi informacje nadchodzące z instalacji technologicznej (sterowniki programowalne, inteligentne regulatory, rejestratory i tp.) wyposażane mogą być w systemy służące do sterowania i wizualizacji pracy obiektu. Pakiet **WIZCON** przeznaczony do realizacji szeroko rozumianych funkcji monitorowania i sterowania dla obiektów energetycznych, przemysłowych i innych, należy do najbardziej znanych na świecie. Uznanawany jest on przez niezależne instytucje badające światowy rynek automatyki przemysłowej (Automation Research Co., USA, 1994 edition) za najlepsze w swych kategoriach w Europie (WIZCON 4.2), czy nawet na świecie (WIZCON/2 dla OS/2). Najnowszy produkt firmy, to **WIZCON wersja 5.0,** którego cechy funkcjonalne predystynują go do zastosowań w dużych, rozproszonych systemach sterowania i zarządzania zakładami przemysłowymi. Jako jedyny tego typu pakiet posiada on certyfikat na zgodność wydany przez producenta systemu operacyjnego OS/2, firmę IBM. Zastosowanie własnych rozwiązań programowych z jednej strony, a dopasowanie do istniejących standardów sieciowych i obsługi baz danych - z drugiej, umożliwiło stworzenie niezwykle nowoczesnego i wysokowydajnego systemu.

Stworzone narzędzie programowe pozwala na tworzenie aplikacji dla praktycznie dowolnego rodzaju i wielkości instalacji obiektowych. Pakiety oprogramowania **WIZCON** pracować mogą na komputerach zgodnych z IBM PC, pod kontrolą różnych systemów operacyjnych (wersja 5.0 dla : OS/2, WINDOWS NT, WINDOWS '95, wersja 4.2 dla MS-DOS). Różnorodność dostępnych modeli **WIZCON**-a zapewnia optymalny dobór wersji pakietu do bieżących potrzeb instalacji obiektowej, z gwarancją możliwości jej rozszerzenia w przypadku konieczności rozbudowy.

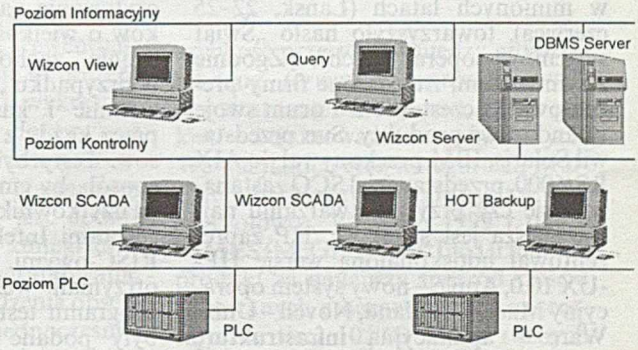
Oprogramowanie **WIZCON** nadzoruje w trybie czasu rzeczywistego pracę obiektowych urządzeń sterujących takich, jak : sterowniki programowalne, regulatory, rejestratory itp. (lista dla całej rodziny przekracza 250 pozycji !), umożliwiając operatorowi obserwację instalacji i oddziaływanie na obiekt za pośrednictwem obrazów animowanych. Dołączenie dowolnego urządzenia spoza listy jest możliwe poprzez napisanie własnego driver'a komunikacyjnego (przy użyciu oprogramowania dostarczanego bez dodatkowych opłat wraz z pakietem głównym). **WIZCON** może więc współpracować z urządzeniami różnych wytwórców (do 16 driver'ów jednocześnie na jednym PC !), tworząc w ten sposób spójny system automatyki. Część rezydentna oprogramowania **WIZCON** odpowiada za odbywającą się w sposób ciągły akwizycję danych (informacje mogą być oznaczane znacznikiem czasu *'time stamp'* z rozdzielczością 1ms, a okres skanowania zmiennych osiąga 50ms). Dane te mogą być archiwizowane na dysku twardym komputera i wyświetlane graficznie na ekranie monitora (monitorów) np. w formie raportów czy wykresów (w tym X-Y). Istnieje również możliwość łatwej wymiany danych z arkuszami kalkulacyjnymi (EXCEL, ACCESS) czy bazami danych (np. DBASE, ORACLE, SQL). Oprogramowanie **WIZCON** umożliwia współpracę wielu stacji operatorskich w lokalnych sieciach komputerowych, zgodnie ze standardami IBM NetBios, IPX/SPX, TCP/IP, dając tym samym szansę wprzęgnięcia instalacji technologicznych w proces zarządzania całym zakładem (patrz opis WIZCON FACTORY).

Do pozostałych funkcji oprogramowania **WIZCON** należą m.in. :

- pełna obsługa stanów awaryjnych systemu sterowania,
- możliwość generowania raportów dowolnego formatu,
- możliwość zapisu stanu systemu i sterowania metodą "receptur",
- automatyczne wykonywanie instrukcji programu napisanego przez użytkownika,
- możliwość realizacji "on-line" algorytmów sterowania napisanych przez użytkownika w dowolnym języku wyższego rzędu z wykorzystaniem bazy danych **WIZCON**-a,
- wielopoziomowy system dostępu i uprawnień operatorskich.

Większość funkcji systemu jest realizowana metodą wyboru opcji z MENU za pomocą klawiszy funkcyjnych. Niezależnie od modelu, udostępnione są wszystkie możliwe do zrealizowania funkcje pakietu **WIZCON** (np. alarmy, raporty, wykresy, driver'y komunikacyjne). Dla zainteresowanych zapoznaniem się z możliwościami oferowanymi przez to oprogramowanie istnieją niezmiernie tanie modele serii EVALUATION.

Wychodząc naprzeciw rosnącym potrzebom przemysłu, firma **PC SOFT Int. Ltd.,** producent **WIZCON**-a, przygotowała i wdrożyła kompleksowe rozwiązanie sieciowego systemu wizualizacji pracy, akwizycji danych i sterowania - **WIZCON FACTORY.** Oparty jest on o pracujący na wielu platformach systemowych (OS/2, WINDOWS NT, WINDOWS '95) pakiet oprogramowania **WIZCON wer. 5.0**. Dzięki pracy w całkowicie 32-bitowym, wielozadaniowym i wielowątkowym systemie operacyjnym, z doskonałą komunikacją wewnętrzną, **WIZCON wer. 5.0** daleko przewyższa inne systemy wizualizacyjne pracujące pod kontrolą WINDOWS czy DOS. **WIZCON** w tej wersji wprowadza milisekundowy czas skanowania zmiennych procesu oraz oddzielenie go od czasu zapisu do zbiorów historycznych, co jest unikalną cechą dla tego typu systemów.



Konfiguracja WIZCON FACTORY

Sieć według koncepcji **WIZCON FACTORY** ma za zadanie umożliwienie wielokierunkowej wymiany danych, wartości, alarmów, wykresów i tp. pomiędzy różnymi jej elementami, jak również zdefiniowanie wspólnych zadań dla wszystkich węzłów (SCADA Node, VIEW Node, SCADA VIEW Node, HOT BACKUP, QUERY Node, WIZCON SERVER, DBMS SERVER, NETWORK SERVER). Pozwala ona na korzystanie z zasobów aktualnych oraz danych historycznych procesu technologicznego przez użytkowników znajdujących się w różnych węzłach sieci. Może być nabudowana na istniejące w firmie systemy sieciowe (oparte o NETBios, TCP/IP czy IPX/SPX), oferując innym służbom zakładu łatwy dostęp do zebranych i przetworzonych danych procesu technologicznego (bazy danych : DBASE, ORACLE, SQL; arkusze kalkulacyjne: np. EXCEL) .

Bliższe informacje techniczne i handlowe oraz wersje demo najnowszej wersji pakietu **WIZCON** otrzymać można w siedzibie jego wyłącznego dystrybutora w Polsce firmie *"SABUR" Sp. z o.o.* , ul. Drużynowa 3A, 02-590 Warszawa, tel./fax (0-22) 44-75-20, 44-63-70.

Najbliższe spotkania targowe z pakietem **WIZCON** :

- SIMMEX (11-15.09.95, " SPODEK ", Katowice, paw.A, st.27),
- SOFTARG (26-29.09.95 MTK, Katowice, paw.8, st. 815) .

ZAPRASZAMY !

0/23/95

informacje

Wiosenne Forum SRSO Łańsk '95

Świat systemów operacyjnych

Wiosennemu Forum Stowarzyszenia Rozwoju Systemów Otwartych, które odbyło się w tym roku później niż w minionych latach (Łańsk, 22–25 czerwca), towarzyszyło hasło „Świat systemów operacyjnych". Zgodnie z tym hasłem, zaproszone firmy prezentowały uczestnikom Forum swoje sztandarowe produkty. Sun przedstawił Solaris, IBM przekonywał do AIX RS/6000, przedstawiciel SCO zastanawiał się czy przy przetwarzaniu najważniejsza jest szybkość, HP zaprezentował udoskonaloną wersję HP-UX 10.0, Apple – nowy system operacyjny Maca – Copland, Novell – UnixWare – aplikacyjną infrastrukturę przedsiębiorstwa, Microsoft zaś zaprezentował strategię systemową firmy.

Najciekawszym jednak wydarzeniem Forum był „Turniej Systemów Operacyjnych". Na podium ustawiono: dwa serwery Optimusa z Pentium i systemem operacyjnym SCO 5.0, serwer Hewletta-Packarda K200 z systemem operacyjnym HP-UX 10.0, dwa serwery SUN (prezentowała firma Comp) – czteroprocesorową maszynę 1000E i SparcStation 20 (jeden procesor HyperSparc) z systemem Solaris 2.4, IBM i Prokom zaprezentowały serwer IBM R39H z systemem operacyjnym AIX 4.10, Inwar – serwer intelowski z systemem operacyjnym SCO 5.0, zaś Intertrading ze Szczecina (dystrybutor amerykańskiej firmy AST) wziął udział w turnieju z intelowskim serwerem AST wyposażonym w system operacyjny UnixWare 2.0.

Maszyny te poddano pięciu testom opracowanym przez Eugeniusza Własiuka – niezależnego konsultanta z Gdańska: obliczenie wartości funkcji tangens w 10 000 punktów (sprawdzanie prędkości koprocesora matematycznego), sprawdzenie wydajności połączenia procesora z RAM, symulacja pracy na bazie danych zorganizowanej na plikach Unix-a (do wykonania 10 tys. operacji zapisywania lub odczytywania na wspólnej puli 100 plików o wielkości do 500 KB), sprawdzenie zachowania się systemów w przypadku „swapowania" oraz powołanie i kasowanie 50 procesów przez każdy z poszczególnych procesów. Testy były uruchomiane w taki sposób, by emulowały pracę od 1 do 64 użytkowników dla serwerów z procesorami Intela i 128 z procesorami RISC-owymi. Uczestnicy turnieju otrzymali wcześniej kod źródłowy programu testującego, w treści testu były podane wartości parametrów oraz zakresy, w których mogły być zmieniane. Ostateczne wartości parametrów wybrali licznie zgromadzeni „kibice". Organizatorzy – a także uczestnicy Turnieju – zastrzegli się, że wyniki nie będą oficjalnie komentowane, głównym bowiem celem imprezy miało być pokazanie w jaki sposób można testować i dostosowywać komputery pod kątem wymagań użytkowników.

W teście, sprawdzającym prędkość koprocesora, lepsze wyniki osiągnęły kompilatory procesorów Intela. W teście sprawdzającym wydajność połączenia procesora z RAM najlepszy okazał się system Solaris na SparcStation 20 z HyperSparc 125 MHz. Pewne trudności sprawił zespołom test symulujący pracę na bazie danych, przyczyniły się do tego podane z widowni parametry testów. Ale w końcu, po pewnych zmianach, prawie wszystkie maszyny uporały się z tym zadaniem. Czwarty test, który polegał na tym, że każdy z procesów przydzielał lub wypełniał 30 obszarów po 32 KB, prowadził do wyczerpania pamięci rzeczywistej i do użycia pamięci dyskowej jako rozszerzenia przestrzeni adresowej komputera. Tutaj wypadły lepiej maszyny Optimusa, Inwara i Sun SparcStation 20. W ostatnim teście zdecydowanie najlepiej wypadły serwery Suna.

Turniej pokazał, że nie ma testów, które by udowodniły, iż istnieje jakaś idealna maszyna pod każdym względem przewyższająca inne. Komputery muszą być dobierane do określonych celów. Widownia oceniała przede wszystkim sprawność zespołów, które na gorąco dostrajały systemy do testów. Wszystkie udowodniły, że potrafią to robić.

Uczestnicy Forum stwierdzili, że jako następny warto przygotować „Turniej baz danych".

Krystyna Karwicka

Książki nadesłane

WYDAWNICTWA NAUKOWO-TECHNICZNE

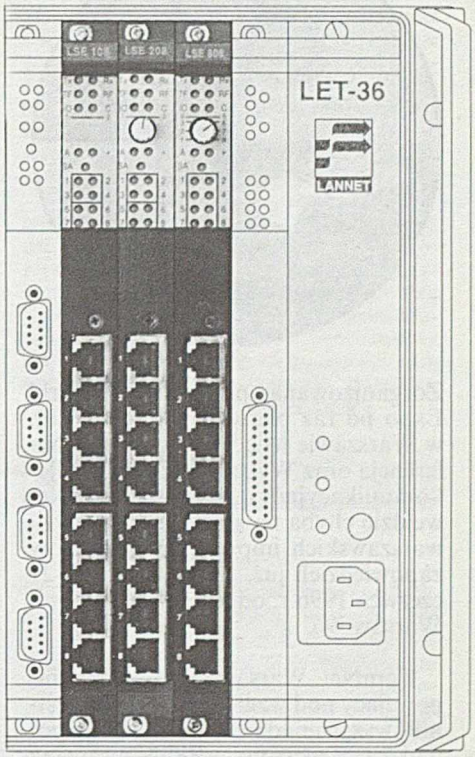
Anatole d'Hardancourt: Microsoft ACCESS dla Windows – zasady użytkowania. Z francuskiego przełożył Marian Łakomy. WNT, Warszawa 1994 r., wyd. 1, s. 136. ISBN 83-204-1694-9

Książka jest doskonałym przewodnikiem dla początkujących użytkowników systemu Access. Do korzystania z niej nie jest potrzebne żadne doświadczenie z bazami danych. Dzięki dokładnym objaśnieniom i prostym przykładom użytkownik może opracować swoją pierwszą bazę danych, nadać arkuszom wprowadzania danych dogodną formę, czerpać informacje z bazy, tworzyć formularze oraz opanować podstawy korzystania z makropoleceń.

System zarządzania bazami danych Access działa na komputerach z procesorem 386SX lub lepszym, z zainstalowanym programem Windows, począwszy od wersji 3.0.

Eberhard Döring: WordPerfect 6.0 dla Windows. Z niemieckiego przełożył Jarosław Cichorski. WNT, Warszawa 1994 r., wyd. 1, s. 172. ISBN 83-204-1774-0

Inteligentne koncentratory MultiNet, czyli ta...ka ryba w Twojej sieci



LANNET Data Communications z Izraela, twórca i trzeci co do udziału na rynku światowym producent komutowanych koncentratorów sieci (ang. *switching hubs*) zorganizowała w czerwcu br. w Warszawie seminarium poświęcone prezentacji swoich produktów. Firma ta obecna już była wcześniej w Polsce przez dystrybutorów.

Główny produkt firmy LANNET to MultiNet – inteligentne, odporne na zakłócenia komutowane koncentratory sieci dla średnich i dużych przedsięwzięć sieciowych o dużym zagęszczeniu gniazd. Dzięki dobrze pomyślanej budowie modularnej są one wdzięcznym tworzywem w rękach zarządców sieci LAN, umożliwiając dużą swobodę w projektowaniu lub modernizacji sieci. W jednej obudowie można zastosować jednocześnie moduły dla wszystkich głównych typów LAN (Ethernet, Token Ring, FDDI, LocalTalk, ATM) w rozmaitych wersjach i dla różnych typów okablowania ze światłowodowym włącznie. MultiNet pozwala na podział LAN na wiele segmentów logicznych i fizycznych oraz na połączenia między sieciami (ang. *bridging i routing*).

Trzon urządzenia stanowi skrzynia zawierająca szybką szynę główną 1.28 Gbps, cztery szyny Ethernetu, dwie Token Ring, cztery LocalTalk i jedną FDDI. Urządzenie występuje w trzech modelach: LET-10, LET-20 i LET-36, mogących pomieścić odpowiednio 10, 20 i 36 modułów połówkowych lub o połowę mniej pełnowymiarowych. W MultiNet uwzględniono systemy pozwalające na niezakłóconą pracę w przypadku awarii, co jest ważną cechą dla strategicznych sieci komercyjnych. Wszystkie układy są dublowane, a uszkodzone moduły można wymieniać bez wyłączania całego urządzenia.

Dla administratorów sieci firma LANNET oferuje szereg udogodnień. MultiMan to wieloplatformowe oprogramowanie do śledzenia pracy i pełnej obsługi koncentratorów MultiNet. Dzięki niemu ma się obraz ruchu w formie statystyki dla całego koncentratora, poszczególnych segmentów logicznych i fizycznych LAN oraz dla pojedynczych gniazd. Za pomocą MultiMan dzieli się LAN na segmenty, przydziela priorytety i definiuje zapasowe łącza. Uzyskuje się też informację o stanie urządzeń i ewentualnych uszkodzeniach. Dane są prezentowane w formie map i wykresów, a zarządzający może dokonywać wszelkich operacji na ekranie graficznym za pomocą myszki. Następnym ciekawym rozwiązaniem jest dodatkowy moduł NMA-RS i zintegrowane z MultiMan oprogramowanie SMONMaster do prowadzenia obserwacji wszystkich gniazd jednocześnie. NMA-RS szczegółowo śledzi ruch na szynie głównej koncentratora w sposób „przejrzysty", czyli bez żadnego wpływu na efektywność urządzenia.

Osiągi MultiNet są godne uznania. Kryteriami branymi pod uwagę w ocenie koncentratorów sieci są opóźnienia na przełączniku (ang. *latency*), szybkość przesyłu pomiędzy gniazdami (ang. *forwarding*), przepustowość bez strat (ang. *throughput*), utrata pakietów i traktowanie zatorów danych. We wszystkich tych kategoriach MultiNet obecnie prowadzi lub jest w ścisłej czołówce. W lutym 1995 r. LAN TIMES (McGraw-Hill, Inc.) uznał ten produkt za najlepszy spośród sześciu testowanych ethernetowych koncentratorów sieci 10/100 Mbps różnych firm. W maju br. MultiNet wraz z modułami Ethernetu LANswitch wygrało z 17 innymi podobnymi urządzeniami w badaniach prowadzonych przez Strategic Networks Consulting (Rockland, MA).

Przemysław Baszkiewicz

W książce opisano edytor WordPerfect 6.0 w środowisku Windows. Omówiono w niej między innymi następujące tematy: instalację programu i jego środowisko, podstawowe funkcje, kontrolę poprawności ortograficznej, dzielenie wyrazów, zarządzanie plikami, definiowanie układu strony, formatowanie akapitów i wierszy, drukowanie dokumentu, opracowywanie stylów, dołączanie grafiki, generowanie spisu treści i i skorowidza, wprowadzanie paginy górnej i dolnej, pisanie formuł i przypisów, opracowywanie makropoleceń.

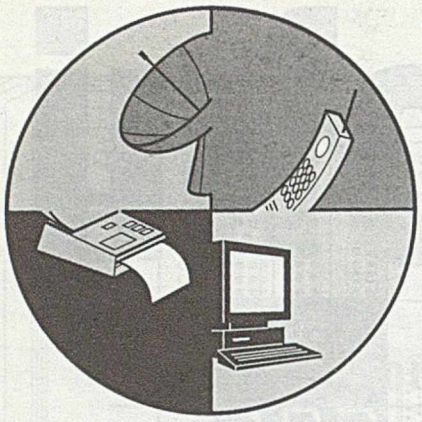
Książka jest przeznaczona dla czytelnika, który chciałby w krótkim czasie przyswoić sobie najważniejsze funkcje tego edytora. Nawet początkujący użytkownik komputera potrafi, po jej przeczytaniu, napisać tekst, sformatować go, zapamiętać i wydrukować.

Wayne Harris: Bazy danych nie tylko dla ludzi biznesu. Z angielskiego przełożył Lech Banachowski. WNT, Warszawa 1994 r., wyd. 1, s. 208.
ISBN 83-204-1678-7

Książka wprowadza czytelnika w świat baz danych, które stanowią kulminacyjne zwieńczenie zastosowania komputerów do gromadzenia i przetwarzania informacji. Bazy danych towarzyszą nam w codziennym życiu.

Na podstawie ORACLE – uniwersalnego systemu tworzącego i obsługującego środowisko relacyjnej bazy danych z poziomu języka SQL – przedstawiono zasady projektowania i budowy baz danych. Wprowadzono także w istotę działania systemu baz danych, w którym wielu użytkowników ma równoczesny dostęp do tych samych danych, jest zapewniona ochrona danych przed ich niewłaściwym użyciem, a po awarii uzyskuje się stracone dane.

Książka jest atrakcyjna zarówno dla czytelnika mającego już doświadczenie w programowaniu, jak i dla czytelnika, który korzystał już z komputera, ale jeszcze sam nie układał programów. (t)



Start imprezy

ComNet Warsaw '95

Ciekawa ekspozycja, perspektywy rozwojowe

Zorganizowana przez IDG World Expo po raz pierwszy w czerwcu br. w Warszawie Międzynarodowa Konferencja oraz Wystawa Techniki Telekomunikacyjnej i Sieciowej ComNet wejdzie chyba na stałe do kalendarza warszawskich imprez. Organizatorzy zapowiedzieli już, że w dniach 19–21 czerwca 1996 r. odbędzie się ComNet Warsaw '96.

ComNet Warsaw '95 nie był imponujący pod względem liczby uczestników: zgromadził na powierzchni ponad 2 tys. metrów w PKiN niespełna 50 wystawców. Ale znalazły się wśród nich największe polskie i zagraniczne firmy z branży telekomunikacyjnej i sieciowej. Integralną częścią wystawy były konferencje, których tematyka dotyczyła technik bezprzewodowych, sieci lokalnych i rozległych, metodyki zarządzania siecią, sieciowych systemów operacyjnych, architektury sieci komputerowych w Polsce. Druga grupa tematyczna dotyczyła polityki Polski w zakresie telekomunikacji oraz uregulowań prawnych związanych z inwestycjami na polskim rynku telekomunikacyjnym. Szczególnie dużym zainteresowaniem cieszyły się seminaria i wykłady poświęcone Internetowi, serwisom on-line, technologii ATM i transmisji głosu w sieci, a więc najnowszym technologiom i usługom telekomunikacyjnym.

Jedną z atrakcji imprezy było bezpośrednie podłączenie stałym łączem 128 kbps kilku stoisk do Internetu (NASK, Prokom, Compaq, Novell). Swoje usługi, związane z Internetem, oferowały firmy ATM, Home Space-Net i Internet Technologies. Doceniając rosnącą popularność Internetu w Polsce, Motorola zapowiedziała wprowadzenie na nasz rynek kompletnych zestawów, umożliwiających podłączenie do Internetu (modem 28,8 kpbs, okablowanie, oprogramowanie i podręczniki).

Mimo dość nikłego zainteresowania publiczności, wszystko wskazuje, że z roku na rok warszawski ComNet będzie jednak odgrywał większą rolę w promowaniu najnowszych technologii i wiedzy na temat sieci i telekomunikacji. Warto przypomnieć, że po raz pierwszy tę bardzo popularną w USA imprezę zorganizowano w Europie Wschodniej przed trzema laty w Pradze, z podobną wówczas do tegorocznej warszawskiej liczbą uczestników. W tym roku w Czechach uczestniczyło już w imprezie 120 firm i 13 tys. zwiedzających.

(kar)

Nowe gniazda UPS-ów

APC dodaje gniazdo rozszerzeń do SmartUPS

Firma American Power Conversion, potentat w produkcji systemów UPS do 5 kVA, zaprezentowała w czerwcu w Warszawie ulepszone modele serii Smart-UPS. Smart-UPS, zaprojektowane z myślą o ochronie złożonych sieci komputerowych, używały dotychczas do łączności ze światem pojedynczego gniazda szeregowego. Teraz producent wprowadził jeszcze tzw. SmartSlot, gniazdo umożliwiające instalowanie dodatkowych kart wewnątrz UPS-a. Ma to uczynić go bardziej uniwersalnym, usprawniając zarówno nadzór nad samym UPS-em, jak i proces automatycznego reagowania sieci w razie awarii zasilania.

SmartUPS to urządzenia typu Line-Interactive o mocach od 250 do 3000 VA, zawierające wszelkie zabezpieczenia przed niejednorodnością zasilania z sieci. Chodzi tu nie tylko o przypadki całkowitego zaniku prądu, ale także o spadki napięcia, nadnapięcia, udary itp. Na przykład, systemy SmartBoost i SmartTrim gwarantują, bez przechodzenia na zasilanie z baterii, napięcie w odpowiednich granicach, nawet jeśli w gniazdku mamy 176 V i 282 V. O stanie rozładowania baterii, obciążeniu, jakości prądu z gniazdka, czy zużyciu akumulatora można dowiedzieć się z kontrolek LED na obudowie i sieciowo. Również przez sieć można UPS skonfigurować. W razie awarii zasilania z gniazdka komputery podłączone do UPS-a otrzymują odpowiedni sygnał, który powoduje powiadomienie użytkownika lub samoczynne bezpieczne wygaszenie systemu wraz z zachowaniem danych. Oprogramowanie obsługi alarmów pracuje w większości popularnych systemów operacyjnych. Karty do SmartSlot mają wspomóc właśnie zastosowania sieciowe.

Oferta SmartSlot obejmuje obecnie trzy karty. Pierwsza z nich, PowerNet, to adapter ethernetowy 10BaseT. Pozwala on przy obsłudze UPS-a korzystać z protokołu SNMP, co jest szczególnie interesujące przy zdalnym zarządzaniu dużą liczbą UPS-ów. Druga karta, UPS Interface Expander, dodaje dwa dodatkowe gniazda szeregowe. Z jednego UPS można więc tym samym kontrolować jednocześnie zasilanie trzech komputerów. Wreszcie ostatnia, Call-UPS II, pozwala podłączyć modem i dokonywać wszelkich operacji przez telefon. Przedstawiciele APC zapewniają, iż asortyment produkowanych akcesoriów zależy jedynie od popytu na poszczególne rozwiązania.

W nowej wersji ze SmartSlot ukażą się modele Smart-UPS 700VA, 1400VA (zwykły i *rackmount*), oraz 3000 VA. Inne modele serii pozostaną niezmienione.

Firma American Power Conversion – producent zasilaczy awaryjnych – zanotowała w ub.r. obroty w wysokości 378 mln USD. Na całym świecie zainstalowano 3 mln UPS-ów produkcji APC. Udział APC w rynku UPS dla zasilaczy o mocy poniżej 5 kVA w Ameryce wynosi 27%, w Europie 21%.

APC od 1 czerwca wprowadziła w Polsce telefoniczny system informowania klientów. Koszty połączenia z tym numerem pokrywa firma. Numer **0 800 25252** działa od godziny 9.00 do 16.00.

Przemysław Baszkiewicz

Novell GroupWise

Główny system obsługi komunikatów

Novell dodał komunikację głosową do GroupWise – systemu obsługi komunikatów. Komunikaty głosowe dołączają do innych typów informacji zintegrowanych w tym systemie, łącznie z pocztą elektroniczną, faksem, elementami terminarza, komunikatami z sieci Internet i formatkami elektronicznymi. Użytkownicy, korzystający z GroupWise, mają bezpośredni dostęp do wszystkich komunikatów. Mogą otrzymywać, odtwarzać, odpowiadać, zapamiętywać i dalej przekazywać komunikaty głosowe ze swojego komputera stacjonarnego, przenośnego lub telefonu.

Uniwersalna skrzynka wejściowa (*Universal in Box*) upraszcza dostęp, korzystanie i zarządzanie przychodzącymi komunikatami danych głosowych, gdyż otrzymuje się je i obsługuje w jednym miejscu. GroupWise wykorzystuje sieciowy system operacyjny NetWare, bazę NetWare Directory Services (NDS) i możliwości administracyjne systemu. Korzystając z NDS administrator tych środowisk może zdecydować – czy zachować stary system komunikacji głosowej, oparty na lokalnej centrali telefonicznej, czy korzystać z możliwości komunikacji głosowej GroupWise lub używać tych obydwu systemów jednocześnie.

W przyszłości system obsługi komunikatów głosowych GroupWise będzie zintegrowany z kontrolą połączeń przez NetWare Telephony Services – platformę integracji sieci lokalnych i central telefonicznych.

Intel

Nowe funkcje antywirusa

Intel włączył do swojego antywirusowego LANDesk Virus Protect 3.0 nowe trzy współpracujące ze sobą elementy, które wykrywają wirusy i chronią przed nimi system komputerowy. Pierwszą barierą jest narzędzie, sprawdzające w czasie rzeczywistym czy kopiowane pomiędzy serwerami i stacjami roboczymi pliki nie przenoszą wirusa. Drugi poziom zabezpieczenia to tzw. *Integrity Shield,* chroniący przed modyfikacjami poszczególne pliki wykonywalne, katalogi i woluminy. Trzecią częścią składową jest pracujący w tle procesor wykorzystywany do kontrolowania serwera poprzez poszukiwanie sygnatur wirusów i testowanie integralności plików.

LANDesk Virus Protect jest obsługiwany przez centralną konsolę, pozwalającą administratorowi łatwo i skutecznie konfigurować i kontrolować pracę pakietu. Jest on także wyposażony w narzędzia raportujące i alarmujące, które ułatwiają zlokalizowanie źródeł infekcji, ich izolację i usunięcie wirusów. Raportowanie umożliwia także śledzenie zdarzeń zachodzących w systemie, identyfikację obszarów systemu nie będących pod ochroną oraz użytkowników rozprowadzających wirusy. Nowy pakiet współpracuje z LANDesk Management Suite, Novell NetWare Management System i Novell ManageWise.

Intergraph

Sześć w jednym

Intergraph Software Solutions wprowadził do sprzedaży nowy pakiet oprogramowania dla inżynierów pod nazwą Design & Engineering Office. Pakiet ten zawiera programy znane już większości użytkowników systemów Intergraphu, ale w połączeniu ich cena jest niższa. Pakiet składa się z następujących programów: IPLOT Client – program kreślarski dla środowiska MicroStation; APLOT Client – program do przygotowywania wydruków, obsługujący pliki w formacie AutoCAD-a; I/RAS B – edytor rysunków rastrowych; DM/ /Librarian – zarządzanie dokumentacją w lokalnych sieciach komputerowych; DM/ /View – program służący do przeglądania dokumentów; ModelView – narzędzie do wizualizacji projektu (rendering i animacje). Pakiet kosztuje od 6500 do 15 900 USD.

Gulipin

Komputery dla inżynierów

Gulipin Computers wspólnie z firmą Mikama z Katowic opracowały serię komputerów osobistych – New Line i Bridge Line. Zostały one zoptymalizowane pod kątem pracy z oprogramowaniem klasy CAD/ /CAM. Zastosowano w nich procesory Intel Pentium o częstotliwości taktowania od 66 do 100 MHz. Karta rozszerzeń jest obsługiwana przez łącza PCI pracujące w 64-bitowym trybie przesyłania danych. Bezpieczeństwo pracy i danych zapewnia zintegrowany UPS 500 VA.

Wszystkie komputery z tych serii mają wbudowane napędy CD-ROM Toshiby o 4,4-krotnej szybkości transmisji i dyski magnetooptyczne Fujitsu o pojemności 230 MB. Grafikę ekranową obsługują karty Mikro Vision wyposażone w 64-bitowe akceleratory grafiki i szybką pamięć VRAM, co umożliwia pracę z dwoma monitorami jednocześnie przy rozdzielczości 1408 × 1024 w trybie High Color. Komputery z tych serii przeszły testy zgodności z oprogramowaniem firm Bentley oraz specjalistycznych aplikacji firmy Intergraph.

Do wymagających użytkowników adresowane są modele wieloprocesorowe Gulipin MP oparte na architekturze Corollary.

Jupiter

Środowisko nowej generacji

Intergraph Software Solution opublikował nową technologię software'ową o nazwie Jupiter, udostępniając ją jednocześnie ponad dwu tysiącom użytkowników, zrzeszonym w IGUG (*International Intergraph Graphics Users Group*). Jupiter jest nowoczesnym środowiskiem dla aplikacji CAD/CAM/CAE/GIS o nowatorskiej architekturze, osadzonym w systemie Windows. Opierając się na opracowanej przez Microsoft specyfikacji COM (*Component Object Model*) środowisko Jupiter znacznie wzmacnia funkcjonalność systemu Windows, rozszerzając jego możliwości o obsługę aplikacji inżynierskich. Pozwala też opracować te aplikacje szybciej i taniej, gwarantując im większą niż dotychczas stabilność i wydajność.

Główne cechy Jupitera, to m.in. biblioteki dla twórców aplikacji, operujące bardziej językiem dyscypliny technicznej, której aplikacja ma służyć, niż typowym językiem programisty, integracja środowiska CAD/CAM/CAE i GIS z aplikacjami biurowymi wyposażonymi w mechanizmy OLE, oraz zwiększenie efektywności tych połączeń dzięki zastosowaniu interfejsu OLE for Design & Modelin. (KK)

Minolta

Małe kopiarki

Minolta rozpoczęła produkcję małych, oszczędnych i wydajnych kopiarek EP 2130 i EP 2131. Obie charakteryzują się dużą szybkością pracy: czas oczekiwania na pierwszą kopię wynosi tylko 6,1 sekundy, na minutę wykonują one 13 kopii formatu A4. Można też wykonywać kopie z papierów o nietypowych formatach lub z folii OHP. Kopiarki mają uproszczone panele operatorskie, są wyposażone w stacjonarną płytę operatorską, co pozwala zaoszczędzić dużo miejsca, zapewnić stabilność minimalizującą cienie i zniekształcenia obrazu przy kopiowaniu z książek i innych grubych oryginałów. Kopiarka EP 2131 ma także nowatorski podajnik oryginałów SHD, który umożliwia automatyczne podawanie do 50 oryginałów A4.

IBM OS/2 Warp Connect

Z dostępem do sieci

IBM wprowadził do sprzedaży nowe oprogramowanie – OS/2 Warp Connect. Jest to sieciowe rozwiązanie dla małych firm lub wydziałów firm dowolnej wielkości, łączące system operacyjny OS/2 Warp z pełnym zestawem możliwości sieciowych. Pozwala ono na pracę w sieciach bez dedykowanego serwera (typu Windows for Workgroups) lub klasyczne rozwiązania z serwerem, zapewnia też dostęp do Internetu i większości popularnych usług sieciowych typu *on-line* z sieci lokalnej lub przez modem. OS/2 Warp Connect zawiera także wbudowany kod pozwalający na bezpośredni dostęp do systemów IBM LAN Server i Novell NetWare bez konieczności zakupu dodatkowego oprogramowania. Jest w pełni zgodny z wszystkimi istniejącymi sieciowymi systemami operacyjnymi, łącznie z Windows NT, Windows for Workgroups, Banyan Vines i Lantastic firmy Artisoft. OS/2 Warp Connect jest wielozadaniowy i odporny na zawieszenie się komputera. Pozwala na uruchomienie aplikacji systemu DOS i środowiska Windows, zawiera też BonusPak, będący zbiorem 32-bitowych aplikacji.

Opracowała
KRYSTYNA
KARWICKA-RYCHLEWICZ

DOSTĘPNY JEST JUŻ NAJszybszy seryjnie produkowany mikroprocesor Intela – Pentium (1110/133 MHz według indeksu iCOMP). Jest to trzeci „członek rodziny" procsorów Pentium wprowadzony na rynek w tym roku. Przeznaczony jest dla serwerów i stacji roboczych o dużej wydajności. Testy wydajności procesora Pentium 133 MHz dały wyniki 155 SPECint92 i 116 SPECfp92 – ponad dwa razy lepsze od wyników uzyskanych przez pierwsze Pentium 60 MHz, wprowadzone na rynek w marcu 1993 r. Nowy procesor jest drugim produktem firmy Intel wykonanym w zaawansowanej technologii 0,35 mikrona przy napięciu 3,3 wolta. W pierwszym kwartale br. zyski ze sprzedaży Pentium przewyższyły zyski ze sprzedaży procesora Intel486.

* * *

3Com PRZEJĄŁ FIRMĘ ACcessWorks Communication, która jest największym dostawcą całej rodziny produktów dla sieci, działających na podstawie cyfrowej techniki wymiany danych ISDN (*Integrated Services Digital Network*) pod wspólną nazwą *„Quick Access"*. Systemy połączeń wykonywane w technice ISDN stanowią jeden z kluczowych komponentów architektury HPSN (*High Performance Scalable Networking*) rozwijaną przez 3Com. HPSN jest strukturą dostarczającą możliwości skalowania wydajności sieci. Do podstawowych urządzeń sieciowych wykorzystujących technologię ISDN należą routery serii NETBuilder II i NETBuilder Remote Office. Firma AccessWorks ma w swojej ofercie serwery zdalnego dostępu AccessBuilder oraz karty sieciowe.

* * *

INTERGRAPH UDZIELAjąc poparcia dla OGF (*Open GIS Foundation* – organizacja zrzeszająca producentów systemów informacji geograficznej) podał do publicznego użytku formaty plików danych własnego oprogramowania MGE (*Modular GIS Environment*). W ten sposób MGE wchodzi w ramy specyfikacji Open GIS, której celem jest wdrożenie ideału otwartości do platform GIS-owskich, umożliwiając użytkownikom pracę na tych samych danych za pomocą aplikacji różnych producentów, działających w różnych środowiskach systemowo-hardware'owych. Poparcie dla OGF to końcowy fragment realizowanego przez Intergraph programu przeniesienia wszystkich aplikacji GIS-owskich do 32-bitowego systemu Windows NT.

* * *

W CENTRUM BADAWCZYM Xerox w Palo Alto w Kalifornii wyprodukowano nowy ciekłokrystaliczny płaski panel, którego rozdzielczość 300 punktów na cal (6,3 mln pikseli) – jest czterokrotnie większa niż w znanych ekranach VGA osiągających do 71 punktów na cal. Jakość wyświetlanego obrazu może być porównywalna jedynie z jakością wydruków z drukarek laserowych. Panel, dostępny w wersji kolorowej i czarno-białej, jest idealny do wyświetlania obrazów, w których drobne szczegóły mają istotne znaczenie, na przykład w medycynie, sztuce, archeologii, kartografii. Rząd Stanów Zjednoczonych przyznał korporacji Xerox specjalny fundusz, który ma służyć wykorzystaniu tej nowej technologii na skalę produkcyjną.

* * *

FIRMA RAD DATA COMmunications, której centrala międzynarodowa mieści się w Tel Awiwie, skupia 14 firm specjalizujących się w rozwiązaniach telekomunikacyjnych do sieci komputerowych. W 1944 r. firma sprzedała 300 tys. urządzeń za 70 mln USD do prawie 70 krajów, a jej całkowite obroty wyniosły 170 mln USD. Około 30% zatrudnionego w RAD personelu zajmuje się badaniami i rozwojem nad takimi technologiami, jak przetwarzanie sygnału mowy i dźwięku, kompresja danych do łączenia sieci lokalnych i oprogramowanie zarządzające sieciami.

* * *

PONAD 500 PROJEKTANtów systemów zarządzania informacją korzysta z pakietu Microsoft Office przy tworzeniu aplikacji zarówno dla dużych korporacji, jak i małych firm. Microsoft Office oferuje efektywne narzędzia, m.in. język makrodefinicji Visual Basic OLE Automation, dzięki czemu jest możliwe wykonanie w krótkim czasie wydajnej aplikacji. To i inne udogodnienia sprawiają, iż Microsoft Office jest najpopularniejszym pakietem do obsługi firmy: do tej pory sprzedano go ponad 9 mln w 24 wersjach językowych.

* * *

CZESKI SPT TELECOM S.A. dostarcza swoim klientom od maja br. usługi według standardu X.400. Rozwiązanie oparto na sprzęcie RISC System/6000, dostarczonym przez IBM w ramach kontraktu obejmującego integrację systemu. System składa się z dwóch komputerów IBM RS/6000 model 58H w klastrze wysokiej dostępności (HACMP – *High Availbility Cluster Multi Processing*), gwarantującym nieprzerwaną pracę przez 24 godz. na dobę. Oprogramowanie zostało stworzone przez firmę ISOCOR. Stopniowo usługa X.400 ma zastąpić dotychczasową sieć teleksową.

* * *

FIRMA FUJITSU WPROWAdza na rynek nową drukarkę laserową LED-PrintPartner 8600. Jej szybkość drukowania wynosi 8 str./min, rozdzielczość druku 600 × 600 pkt/cal. Wyposażona jest w procesor SPARC Lite oraz w 2 MB pamięci RAM (rozszerzalne do 10 MB typowymi układami SIMM 32 bit). Drukarka jest kompatybilna z HP LaserJet IV, ma wiele czcionek bitmapowych i skalowalnych oraz liczne alfabety narodowe, w tym standardy polskie. Jako opcja pojawi się moduł z PostScriptem oraz dodatkowe interfejsy szeregowe RS 232, AppleTalk, Ethernet A i Ethernet B. Firma INTEL – dystrybutor Fujitsu w Polsce – oferuje nową drukarkę już od sierpnia. **(kar.)**

Zaprosili nas...

Stowarzyszenie Rozwoju Systemów Otwartych na spotkanie informujące o kolejnym Wiosennym Forum w Łańsku, organizowanym w tym roku pod hasłem: Świat Systemów Operacyjnych. W bieżącym roku SRSO planuje jeszcze spotkanie nt. organizacji danych – Metodologia oraz organizacji struktur – Przedsięwzięcie. **(t)**

Ministerstwo Przemysłu i Handlu oraz Zarząd Targów Warszawskich Biura Reklamy SA na uroczyste otwarcie III Międzynarodowych Targów Zastosowań Informatyki w Przemyśle i CAD/CAM '95.

Autodesk do odwiedzenia stoiska firmy System 3000 na III Międzynarodowych Targach Zastosowań Informatyki w Przemyśle i CAD/CAM '95.

Biuro Techniczne i Konferencyjne PAI EXPO we współpracy z **IDG Poland** na Pierwszą Międzynarodową Konferencję i Wystawę Techniki Telekomunikacyjnej i Sieciowej ComNet Warsaw '95, odbywającą się w Pałacu Kultury i Nauki w Warszawie.

Compaq Computer Sp. z o.o. na śniadanie prasowe z okazji wystawy ComNet '95 oraz na pokaz dostępu i wykorzystania możliwości Internetu na komputerach Compaqa.

ATM do odwiedzenia stoiska w trakcie trwania wystawy ComNet '95.

RAD Data Communications na śniadanie prasowe podczas ComNet '95 oraz zwiedzenie swojego stoiska.

IDG Word Expo i NOKIA na przyjęcie do hotelu Marriott z okazji wystawy ComNet '95.

Microsoft Sp. z o.o. na konferencję poświęconą programowaniu w środowisku klient-serwer.

IBM Polska na spotkanie, podczas którego zaprezentowano komputery AS/400 z najnowszą generacją procesorów PowerPC.

Elbatex GmbH na prezentację nowych propozycji firmy SEI/Elbatex w sieci Internet.

ABC Data na spotkanie informacyjne firmy Allied Telesyn, światowego dostawcy produktów dla systemów sieciowych LAN. Przedstawiono uwarunkowania rynkowe obejmujące rolę hubów wieżowych, nowe technologie Ethernet oraz obowiązujące normy europejskie i międzynarodowe.

IBM na seminarium poświęcone zagadnieniom systemów otwartych, prowadzone przez Terry Keene'a – wybitnego światowego konsultanta z firmy Enabling Technology Group Inc. ze Stanów Zjednoczonych. Omówiono trendy w rozwoju różnych linii procesorów i ich implementacje jedno- i wieloprocesorowe oraz przeanalizowano różne rozwiązania systemów operacyjnych.

OPTIMUS SA na uroczyste podpisanie umowy Value Added Reseller między AT & T Network System a OPTIMUSEM. **(t)**

Perfect-Ekspert AITECH

20 LAT DOŚWIADCZEŃ WDROŻENIOWYCH

Wysoka jakość

Wysoka jakość

„Polifarb" Cieszyn S.A.
Zakłady Rowerowe „ROMET"
Krywałd „ERG" S.A.
Elektrownia Rybnik Huta Łaziska
Fabryka Osi Napędowych - Radomsko
Zakłady Tłuszczowe „BIELMAR"
Elektromontaż 3 Huta Bankowa
Zakłady Metalurgiczne „Trzebinia"
Walcownia Metali „Dziedzice" Huta Szkła „Kunice"
POLAM Kostuchna Zakłady Gumowe Górnictwa
Huta 1-go Maja
IBM PC IBM PC IBM PC IBM PC
ODRA 1305

AS/400 AS/400 AS/400 AS/400

1995 Zakres funkcjonalny

Sieciowy i zintegrowany system informatyczny, głównie dla dużych firm produkcyjnych i handlowych. Przystosowany do elektronicznej wymiany dokumentów (EDI - zamówienia, faktury, WZ, itp.), wielowalutowości, denominacji złotego oraz rachunkowości zarządczej. Wyposażony w system ochrony danych generowanych wewnątrz i na zewnątrz przedsiębiorstwa. Współpracujący z systemem ekspertowym ISAF (Inteligentny System Analiz Finansowych) wspomagający **podejmowanie decyzji finansowych na średnich i wyższych szczeblach zarządzania.**

AMAPOVELL

System Informacji Kierownictwa	Inteligentny System Analiz Finansowych	System Finansowo-Kosztowy	System Wyposażenie
System Obrotu Towarowego i Fakturowania	System Obrotu Materiałami i Ewidencji Zakupu	Rozliczenie Zakupu Materiałów	System Środki Trwałe
System Kadry - Płace	System Analiz Marketingowych	System Technicznego Przygotowania Planowania i Rozliczania Produkcji	EDI TMW Ultra - Perfect

1987 Zakres funkcjonalny

Oferujemy kompleksowe usługi w zakresie:
- doradztwa organizacyjnego, merytorycznego, technicznego, consulting,
- organizacji, projektowania i wykonawstwa innych systemów informatycznych,
- wdrażania, konserwacji i modernizacji systemów informatycznych,
- projektowania, wykonawstwa dowolnego typu sieci komputerowych, z uwzględnieniem już istniejących rozwiązań,
- dostawa sprzętu komputerowego i sieciowego,
- szkoleń użytkowników w zakresie operatora i administratora systemu AS/400, jak również obsługi pakietów programowych PC Support/400 lub Client Access/400 przeznaczonych do współpracy z PC.

JUNISOFTEX

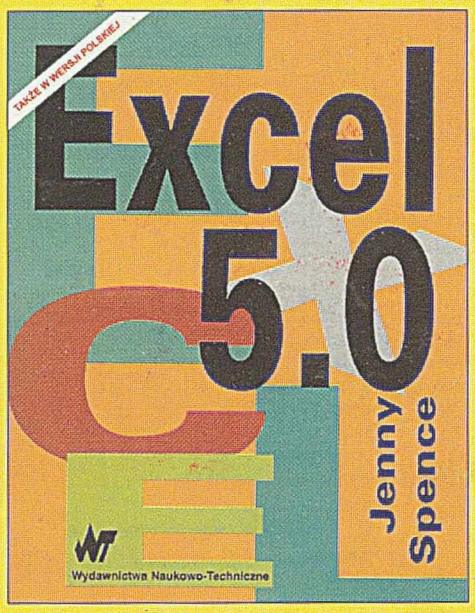
JUNISOFTEX

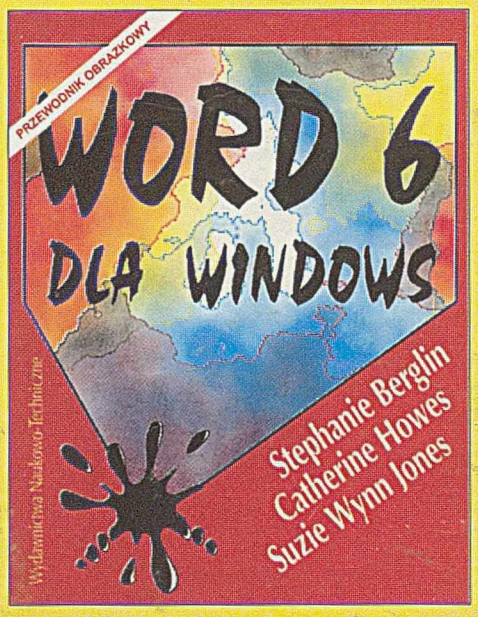
44-100 Gliwice, ul. Kościuszki 1 c, box 507
Tel./fax (0-32) 31-51-31, 31-57-00
38-80-52, 38-80-54

Wydawnictwa Naukowo-Techniczne
00-048 Warszawa, Mazowiecka 2/4

POLECAJĄ! POLECAJĄ! POLECAJĄ!

- Berglin S., Howes C., Jones S. - **Word 6 dla Windows** (przewodnik obrazkowy) - 25,00 zł
- Dunn S., Bermant Ch., Berst J. - **Windows 3.1 - 101 wskazówek i trików** - 5,00 zł
- Harris W. - **Bazy danych nie tylko dla ludzi biznesu** - 12,00 zł
- Hertz J., Krogh A., Palmer R. G. - **Wstęp do teorii obliczeń neuronowych** - 12,00 zł
- Koblitz N. - **Wykład z teorii liczb i kryptografii** - 15,00 zł
- Nance B. - **Windows for Workgroups** - 21,00 zł
- Rodgers U. - **ORACLE®. Przewodnik projektanta baz danych** - 25,00 zł
- Software W. - **SQL. Język relacyjnych baz danych** - 14,00 zł
- Southerton A., Perkins E. jr. - **Słownik poleceń systemów UNIX i X** - 43,00 zł
- Spence J. - **Excel 5** - 17,80 zł





Każdą z tych książek można nabyć za zaliczeniem pocztowym. Przy zamówieniu o wartości przekraczającej 25,00 zł koszty przesyłki pokrywa WNT, a dołączenie kuponu upoważnia do uzyskania 10% rabatu.

Nasz adres: Dział Handlowy WNT, skr. poczt. 359, 00-950 Warszawa, tel./fax 26-82-93

KUPON Informatyka 1/95

0/21/95